



Object-Oriented Language : CORBA



- See also ► [CORBA ORBs](#) ► [Enterprise JavaBeans](#)

- **Intro**

- *CORBA* (Common Object Request Broker Architecture), is a distributed object architecture that allows objects to interoperate across networks regardless of the language in which they were written or the platform on which they are deployed.
- *CORBA* allows developers to write applications that are more flexible and future-proof, to wrap legacy systems, and to code in the language they know best.
- The Object Request Broker (ORB) is the middleware that handles the communication details between the objects. The CORBA 2.0 standard, adopted in December of 1994, defines true interoperability by specifying how ORBs from different vendors can communicate using a common protocol.

- **Contents**

- [Start here](#)
- [Demos and Examples](#)
- [Fun](#)
- [Central Sites](#)
- [Link Collections](#)
- [Tutorials](#)
- [FAQs](#)
- [References](#)
- [Standards](#)
- [General Newsgroups](#)
- [Mailing Lists](#)
- [General Articles](#)
- [Special Articles](#)
- [Interoperability](#)
- [Bibliographies](#)
- [Books](#)
- [Magazines](#)
- [Organizations](#)
- [Projects](#)
- [Conferences / Workshops](#)
- [Other Resources](#)
- [Products / Companies](#)

- **Other pages**

- [Distributed Objects & Components](#)
- [COM/COM+](#)
- [Java ...](#)
- [Servers](#)
- [ADA](#)
- [COBOL](#)
- [Java ...](#)
- [Patterns](#)
- [Tcl/Tk](#)
- [UML](#)



Title: Application Development

Document Number: GG24-4481-00

Build Date: 12/09/94 12:58:50 **Build Version:** 1.2

Book Path: /home/webapps/epubs/htdocs/book/gg244481.bo

CONTENTS Table of Contents

[\[Summarize\]](#)

TITLE	Title Page
NOTICES	Notices
EDITION	Edition Notice
ABSTRACT	Abstract
CONTENTS	Table of Contents
FIGURES	Figures
TABLES	Tables
FRONT_1	Special Notices
PREFACE	Preface
PREFACE.1	How This Document Is Organized
PREFACE.2	Related Publications
PREFACE.3	Other Publications
PREFACE.4	Acknowledgments
1.0	Chapter 1. Introduction To OSF/DCE
1.1	DCE Design Considerations
1.2	Distribution Possibilities
1.3	Open Blueprint
1.4	Client/Server Model
1.5	DCE Remote Procedure Call (RPC)
1.6	DCE Threads
1.7	Distributed Services
1.7.1	Directory Service
1.7.2	Security Services
1.7.3	Time Services
1.7.4	Distributed File System
1.7.5	DCE Cell
1.8	MVS/ESA OpenEdition
2.0	Chapter 2. Development Process and Interface Development
2.1	DCE Application Development Steps
2.1.1	The DCE Application Development Steps
2.2	Application Building
2.3	Interface Development
2.4	Server Development
2.5	Client Development
2.6	Generating a UUID and IDL File

2.6.1	<u>Interface Header</u>
2.6.2	<u>Interface Body</u>
2.6.3	<u>Defining the .acf File</u>
3.0	<u>Chapter 3. Server</u>
3.1	<u>Implementing Remote Procedures</u>
3.2	<u>Initializing Server Code</u>
3.3	<u>Registering the Interface</u>
3.3.1	<u>Entry Point Vector</u>
3.3.2	<u>rpc server register if()</u>
3.3.3	<u>Protocol Sequences</u>
3.3.4	<u>rpc server use all protseqs()</u>
3.3.5	<u>rpc server use protseq()</u>
3.3.6	<u>rpc server use protseq ep()</u>
3.4	<u>Advertising the Server</u>
3.4.1	<u>Endpoints</u>
3.4.2	<u>Endpoint Mapper Daemon, rpcd</u>
3.4.3	<u>Steps to Advertise the Server</u>
3.4.4	<u>rpc server inq bindings()</u>
3.4.5	<u>rpc ep register()</u>
3.4.6	<u>rpc ns binding export()</u>
3.5	<u>Listening for RPC Calls</u>
3.6	<u>Error Handling</u>
3.7	<u>Example of Server Initialization Code</u>
4.0	<u>Chapter 4. Client</u>
4.1	<u>Binding</u>
4.1.1	<u>Running an Application Using Explicit Binding Management</u>
4.1.2	<u>Implementing Explicit Binding Using CDS</u>
4.1.3	<u>Running Implicit Binding Using CDS</u>
4.1.4	<u>Implementing Implicit Binding Using CDS</u>
4.1.5	<u>Implementing Automatic Binding</u>
4.1.6	<u>Running With String Binding</u>
4.1.7	<u>Implementing String Binding</u>
4.2	<u>Running with a Well-Known Endpoint</u>
4.2.1	<u>Implementing Well-Known Endpoints</u>
4.2.2	<u>Running in a OpenEdition HFS Environment</u>
4.3	<u>Creating Files for Your Application</u>
5.0	<u>Chapter 5. Compiling and Link Editing</u>
5.1	<u>Compiling the Interface with the IDL Compiler</u>
5.1.1	<u>Compiling the IDL</u>
5.2	<u>Compiling the Client and Server Programs</u>
5.3	<u>Link-Editing Your Application</u>
6.0	<u>Chapter 6. Security</u>
6.1	<u>Concepts of DCE Security</u>
6.1.1	<u>Authentication Levels</u>
6.1.2	<u>Communication Protection Levels</u>
6.1.3	<u>Authorization</u>
6.1.4	<u>DCE Authorization</u>
6.1.5	<u>DCE Access Control Lists</u>
6.1.6	<u>Name-Based Authorization</u>
6.2	<u>Example</u>
6.2.1	<u>Description of the Sample Application</u>
6.2.2	<u>Interface Definition</u>
6.2.3	<u>Attribute Configuration File</u>
6.2.4	<u>Server Implementation</u>
6.2.5	<u>Server Module</u>
6.2.6	<u>Manager Module</u>
6.2.7	<u>Security Module</u>
6.2.8	<u>Client Module</u>
6.2.9	<u>Running the Example Application</u>
6.3	<u>DCE Security and RACF</u>

6.4	<u>A DCE Application Server on MVS</u>
6.4.1	<u>Application Resources Controlled by DCE Security</u>
6.4.2	<u>Application Resources Controlled by RACF</u>
6.5	<u>A DCE client on MVS</u>
6.5.1	<u>Encrypted DCE Password in RACF Database</u>
6.5.2	<u>SAVEPW Command</u>
6.5.3	<u>DCE System Secret Key Utility</u>
7.0	<u>Chapter 7. Threads and Data</u>
7.1	<u>Threads</u>
7.2	<u>Data Handling</u>
7.2.1	<u>Conformant Arrays</u>
7.2.2	<u>Unions</u>
7.3	<u>Dynamic Storage allocation</u>
8.0	<u>Chapter 8. DL/I and DB2 Server Considerations</u>
8.1	<u>Ensuring the Integrity of Data Base Updates</u>
8.2	<u>DB2 Server Overview</u>
8.2.1	<u>Call Attach Facility (CAF) Overview</u>
8.2.2	<u>Call Attach Facility Usage</u>
8.2.3	<u>DB2 Manager Code</u>
8.2.4	<u>DB2 Functional Code</u>
8.3	<u>DL/I Access from the DCE Server to DL/I Databases</u>
8.3.1	<u>Implementation of a CCTL in a DCE Server</u>
8.3.2	<u>Use of the CCTL Interface in the DCE Managers</u>
8.4	<u>Split-up of the DCE Server</u>
8.5	<u>Summary on DL/I and DB2 Access from Manager Code</u>
9.0	<u>Chapter 9. IMS Dependant Regions as DCE Clients</u>
9.1	<u>BMP Main Program in C</u>
9.1.1	<u>C Subroutine for BMP C Main</u>
9.2	<u>BMP Main Program in PL/I</u>
9.2.1	<u>C Subroutine for BMP PL/I Main</u>
9.3	<u>BMP Main Program in COBOL</u>
9.3.1	<u>C Subroutine for BMP COBOL Main</u>
10.0	<u>Chapter 10. DB2 Dynamic SQL Server</u>
10.1	<u>Dynamic SQL Overview</u>
10.2	<u>Dynamic SQL IDL</u>
11.0	<u>Chapter 11. Configuring an Application Client and Server</u>
11.1	<u>Steps to Configure an Application Client and Server</u>
11.2	<u>Configuring an Application Server</u>
11.3	<u>OpenEdition DCE Environment Variables</u>
A.0	<u>Appendix A. JCL and Source Listings for MATHX and SBIND Examples</u>
A.1	<u>MATHX JCL and Source Listings</u>
A.1.1	<u>Compile MATHX Client and MATHX Client Stub</u>
A.1.2	<u>Compile MATHXS Server and MATHX Server Stub</u>
A.1.3	<u>Compile MATHXM Manager</u>
A.1.4	<u>LINKEDIT MATHX Client</u>
A.1.5	<u>LINKEDIT MATHX Server</u>
A.1.6	<u>RUN MATHX Client</u>
A.1.7	<u>RUN MATHX Server</u>
A.2	<u>Source for MATHX Example</u>
A.2.1	<u>Client Source (MATHXC)</u>
A.2.2	<u>Server Source (MATHXS)</u>
A.2.3	<u>Manager Source (MATHXM)</u>
A.2.4	<u>MATHX OUTPUT</u>
A.3	<u>SBIND JCL and Source Listings</u>
A.3.1	<u>Compile SBIND Client and SBIND Client Stub</u>
A.3.2	<u>Compile SBIND Server and SBIND Server Stub</u>
A.3.3	<u>Compile SBINDM Manager</u>
A.3.4	<u>LINKEDIT SBIND Client</u>

A.3.5	<u>LINKEDIT SBIND Server</u>
A.3.6	<u>RUN SBIND Client</u>
A.3.7	<u>RUN SBIND Server</u>
A.4	<u>Source for SBIND Example</u>
A.4.1	<u>Client Source (SBINDC)</u>
A.4.2	<u>Server Source (SBINDS)</u>
A.4.3	<u>Manager Source (SBINDM)</u>
A.4.4	<u>SBIND OUTPUT</u>
<hr/>	
B.0	<u>Appendix B. DCE Security Example</u>
B.1	<u>Server Module C Source</u>
B.2	<u>Manager Module C Source</u>
B.3	<u>Security Module C Source</u>
B.4	<u>Client Module C Source</u>
B.5	<u>Common Header File</u>
B.6	<u>MBOXLIB File</u>
B.7	<u>Message REXX exec</u>
B.8	<u>Batch JCL to Run Messagebox Server</u>
C.0	<u>Appendix C. MATHX Source (Well-Known Endpoint)</u>
C.1	<u>MATHX Client with Well-Known Endpoint</u>
C.2	<u>MATHX Server with Well-Known Endpoint</u>
C.3	<u>Well-Known Endpoint OUTPUT (printf statements)</u>
C.4	<u>Makefile for MATHX OE Example</u>
D.0	<u>Appendix D. IMS and DB2 Examples</u>
D.1	<u>DB2</u>
D.1.1	<u>IDL</u>
D.1.2	<u>Header</u>
D.1.3	<u>Client Source Code</u>
D.1.4	<u>Server Source Code</u>
D.1.5	<u>Manager Source Code</u>
D.1.6	<u>Function Source Code</u>
D.2	<u>DL/I</u>
D.2.1	<u>IDL</u>
D.2.2	<u>Header</u>
D.2.3	<u>Client Source Code</u>
D.2.4	<u>Server Source Code</u>
D.2.5	<u>Manager Source Code</u>
D.2.6	<u>Function Source Code</u>
E.0	<u>Appendix E. IMS Dependent Region as DCE Client Source</u>
E.1	<u>C Source</u>
E.1.1	<u>C Main</u>
E.1.2	<u>C Subroutine for BMP Source</u>
E.2	<u>PLI Source</u>
E.2.1	<u>PLI Main</u>
E.2.2	<u>PLI Subroutine for BMP Source</u>
E.3	<u>COBOL Source</u>
E.3.1	<u>COBOL Main</u>
E.3.2	<u>COBOL Subroutine for BMP Source</u>
F.0	<u>Appendix F. Dynamic SQL Server</u>
F.1	<u>IDL</u>
F.2	<u>Header</u>
F.3	<u>Client Source Code</u>
F.4	<u>Server Source Code</u>
F.5	<u>Manager Source Code</u>
G.0	<u>Appendix G. Installation Prerequisites</u>
G.1	<u>Required PTFs</u>
G.2	<u>Prerequisites for OpenEdition DCE Application Support 1.1.0</u>
INDEX	<u>Index</u>

COMMENTS

ITSO Technical Bulletin Evaluation

RED000



© Copyright IBM Corp. 1994

IBM Library Server® Copyright 1989, 2003 IBM Corporation. All rights reserved.



1.0 Chapter 1. Introduction To OSF/DCE

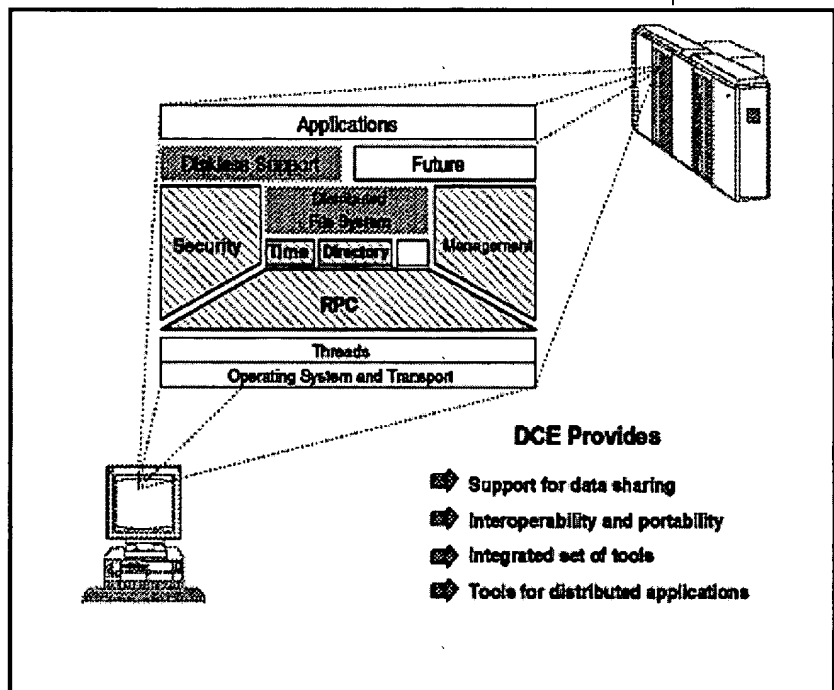


Figure 1. OSF/DCE

Open Software Foundation (OSF) Distributed Computing Environment (DCE) is composed of a set of services that support the development, use, and maintenance of distributed applications. The services, which are also called DCE technology components, fall into two generic categories:

- Programming services
- Distributed services

DCE threads and RPC are programming services that include libraries that implement application programming interfaces (APIs) and program development tools.

The remaining DCE technologies are distributed services: the directory server, the time server, the security server, and the file server. They consist in part of a *daemon*, or server process, that runs continuously on a machine and responds to requests that are sent over the network. The distributed services are equipped with administrative components to manage

the service. They also have APIs through which programmers can access the server.

Application programmers deal mostly with the programming services. Although the distributed services are accessed through their APIs, the programmer usually uses the distributed services indirectly - through RPC, which in turn uses the distributed services APIs.

This document describes how you can develop distributed applications that makes use of the DCE RPC and Threads services.

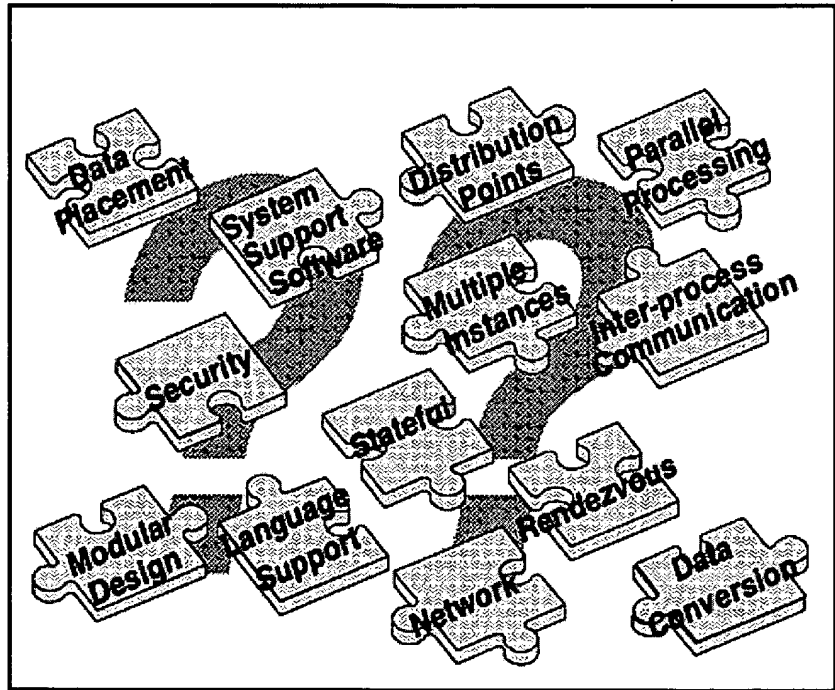


Figure 2. Client/Server Design Issues

Subtopics:

- 1.1 DCE Design Considerations
- 1.2 Distribution Possibilities
- 1.3 Open Blueprint
- 1.4 Client/Server Model
- 1.5 DCE Remote Procedure Call (RPC)
- 1.6 DCE Threads
- 1.7 Distributed Services
- 1.8 MVS/ESA OpenEdition





1.1 DCE Design Considerations

Building software always seems harder than it ought to be. It takes longer than expected, the functionality and performance are not as expected and the resulting product is not easily changed. A software "crisis" has been going on for 25 years. The crisis has now reached a point where it is impacting the life of the enterprise. With a two to three-year backlog of applications to be developed and a two to three-year software development cycle, information systems are not able to provide the competitive advantages to the business that the underlying technology promises.

In addition, the applications that are needed in the 1990s are becoming much more complex. Distributed applications are now needed. These distributed applications involve multiple platforms from multiple vendors using many software offerings. One of the major objectives of OSF/DCE is to address some of these issues. Let's briefly go through some of the issues associated with client/server design.

- Distribution Points

Perhaps one of the first questions that comes to mind when thinking of distributed applications is where will the application be partitioned? Although there appears to be an infinite number of possibilities, developers are finding several distribution models helpful in designing their applications. We will discuss these models on Figure 3 in topic 1.2

- System Software Support

Key to the application design process is the base upon which application rests. That is, what type of system support is available. The type of things that have to be dealt with here are multi-tasking, scheduling, resource allocation, locking and recovery. Although the technology associated with multi-tasking is generally thought of as an operating system function, in the client/server environments application programmers may have to deal with these technologies also. The issue becomes particularly crucial when writing servers that support many clients. Many of the complexities of distributed application development can be handled by system software. One of the main purposes of DCE is to provide a layer of system software to ease the application development process.

- Inter-process Communication

The most popular forms of inter-process communication are conversation, remote procedure call, and message queueing. Although client/server applications can be written using any of these inter-process communication technologies, the remote procedure call is probably the most natural. We will, of course, be focusing on the remote procedure call in this presentation.

- Rendezvous

One of the obvious concerns for client/server applications is how the

client is going to find the server that it needs. That is, how they are going to find each other or rendezvous. DCE provides many services for addressing these concerns with the directory services, but the application developer has to invoke the services. We will see later in this presentation how this affects the server design and in some cases the client design.

- Security

As we have discussed before, security in a client/server environment is more complex than in a single-host environment. The risks are greater and the solutions are more involved. The application design issue is the potential risk associated with a security breach versus the overhead of the security protection. The DCE security services provide a range of security protection that an application can use.

- Modular Design

Modular design techniques are important in any software design. Client/server design requires a stricter adherence to good modular design practices. The interface between modules must be more exact than in centralized designs. In centralized design many modularity problems can be resolved by shared global or external data; this is not possible in a client and server environment. One of the motivating factors in the movement toward client/server is the reuse of servers in application construction. The reuse of a server is dependent on how coherent its design is.

- Parallel Processing

One of the appealing characteristics of client/server applications is that they can use many processors to accomplish a given task. Parallel processing application requirements impact the way both the client and server are designed. DCE provides multi-threading support for parallel processing. We will see later how this is accomplished.

- Language Support

Which programming language is used is an important consideration in application design. DCE is written in C and designed for use by C programmers. The data representation in DCE is patterned after C, the client/server interface assumes a C development process; and the application programming interfaces assume a C caller. It isn't clear how easily another programming language such as COBOL or PL/1 will work directly with DCE.

- Stateful/Stateless Servers

If a server can be called repeatedly by a specific client, it may make sense to have the server remember who has called it before and with what results. The DFS is an example of a stateful server that remembers that the client has opened a file when it comes to make a read request and so on. Care will have to be taken over the design of such a server where the recovery scenarios could be quite complex.

- Multiple Instances

Servers could be replicated many times. You may wish to replicate servers for performance or availability reasons. Are these going to provide identical services or are they going to be different? How will a client choose between multiple instances? Also, servers must be able to support several clients at the same time. How will this be handled?

- Data Conversion

There are many ways of representing data in a computer. It seems like every possible alternative has been tried by some vendors. In centralized processing, the different data types are just an idiosyncrasy of the platform. In a client/server environment, the different data types and format becomes a major design issues. The kinds of problems that must be addressed are ASCII/EBCDIC, big-endian/little-endian, floating point formats, code pages, and so on.

- Data Placement

Where data is placed in the network can become a significant performance and availability issue. If the data and the application that uses the data are in different places, the network can bottleneck and long delays can result. Data that is widely used can shut down the complete environment when not available. Distributed design can be driven by data placement. DCE will not tell the designer where to put the data, but when the decision is made, it can help with the solution. Typical solutions involve techniques of fast data movement and replication of the data.

- Network

There are many kinds of telecommunication networks. Many enterprises have several networks. Each network's software provides a programming interface that is different. This means, for example, with a TCP/IP network the sockets or streams interface is used; for a SNA network, the programmer may use the LU 6.2 protocol. In dealing with these protocols, the programmer must be familiar with the network addressing schemes, recovery processes, message buffering, and so on.



© Copyright IBM Corp. 1994

IBM Library Server® Copyright 1989, 2003 IBM Corporation. All rights reserved.



1.2 Distribution Possibilities

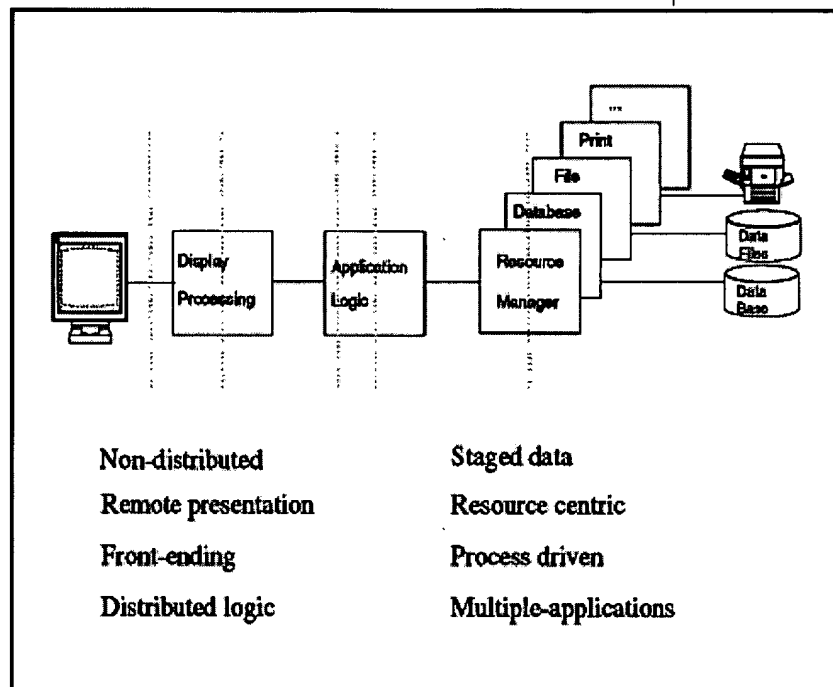


Figure 3. Distribution Possibilities

What are the methods of splitting programs in order to distribute them? Theoretically, the application designers can divide their program any place in the execution sequence. That is what is implied by Figure 3. IBM did a study of those customers developing distributed applications to see where the splits actually take place. We saw a pattern in these applications. Basically, they fell into one of eight groups: non-distributed, remote presentation, front-ending, distributed logic, staged data, resource centric, process driven, and multi-application. The non-distributed applications are not of interest to this presentation and are included only for contrast with the distributed designs. Let's look at the others:

- Remote Presentation

Remote presentation could be represented in Figure 3 as a line to the far left of the Display Processing box. There is no application code stored on the workstation. The presentation service is distributed,

part on the workstation and part on the host. This type of arrangement uses interface management systems such as X-windows. This approach is often used as the simplest way to attach a workstation to a host. Typically users run personal productivity software on the workstation and use multiple windows with terminal emulators to access several hosts.

- Front-Ending

The front-ending design differs from remote presentation because the workstation has front-end application logic. The front-end application logic may be used to simply transform the user's interface or to integrate information from other sources. In either case, the display image is generated twice - once at the host and once at the workstation. Easel** for OS/2 is an example of this type of design.

- Distributed Logic

In the distributed logic design, the workstation and host application components interact with each other directly by program-to-program communication. The conversational, remote procedure call, and the message queueing models are used. This type of design is usually a new application in contrast to front-ending. An important consideration for this design is that the stored data is centralized on the back-end system. The business needs of the enterprise may dictate centralized data for sharing, integrity, or security reasons.

- Data Staging

Data staging typically involve a 3-tiered design approach. The master copy of the data remains on a regional back-end system, while snapshots of the data are staged to local or departmental systems. This provides performance and availability improvements for the users, performance improvements through quicker response time, and less demand on the network. Availability is provided through no single source of failure.

- Resource Centric

Remote resources are accessed through programming interfaces used for local resources in the resource centric design. A well-known example of this is the Network File System. A resource-centric design works best when the amount of data accessed is easily supported by the available communication bandwidth. If large amounts of data have to be moved, significant performance problems can result.

- Process-driven

The process-driven design may be characterized by a step-wise execution of the application. Each application may be viewed as a job step in a higher-level process that is designed to accomplish a business-oriented task. Typically, a workflow manager regulates the overall execution of the business process. In this type of design, a message-style service often seems to be preferred over conversational or RPC.

- Multiple Application

The multiple-application design has application logic and stored data split apart and distributed. The data is private to the distributed application and is not accessible by applications on other nodes. The various pieces of the application are active at the same time and engage in simultaneous communication with each other. This design is suggestive of an object-oriented style where each application

encapsulates its data.

So where does DCE fit with these design models? Resource-centric design with NFS made RPC a common practice in the UNIX world. It is likely that system vendors will continue to exploit RPC to provide distributed services. Remote presentation and front-ending will also be supported by software vendor products that will use RPC. It is likely that process-driven applications will prefer the message queueing inter-process communication technology.

Application programmers will be using DCE for writing application that fit the distributed logic, data staging, or multiple-application design models.

The distributed-design models discussed this section are based on a paper written by Dr. John Shedletsy and John Rofrano. The complete results the customer study is in *Application reference design for distributed systems*, IBM Systems Journal Vol.32 No.4, 1993.



© Copyright IBM Corp. 1994

IBM Library Server® Copyright 1989, 2003 IBM Corporation. All rights reserved.



1.4 Client/Server Model

A useful model for implementing distributed application is the *client/server* model. In this model, a distributed application consists of two parts: a client program and a server program. The two programs are usually running on different systems attached to a network, and talking with each other using its unique protocols. The client's role is to ask the server part to carry out user's requests on behalf of its user. The server then fulfills the client's request. [Figure 5](#) illustrates the client/server model.

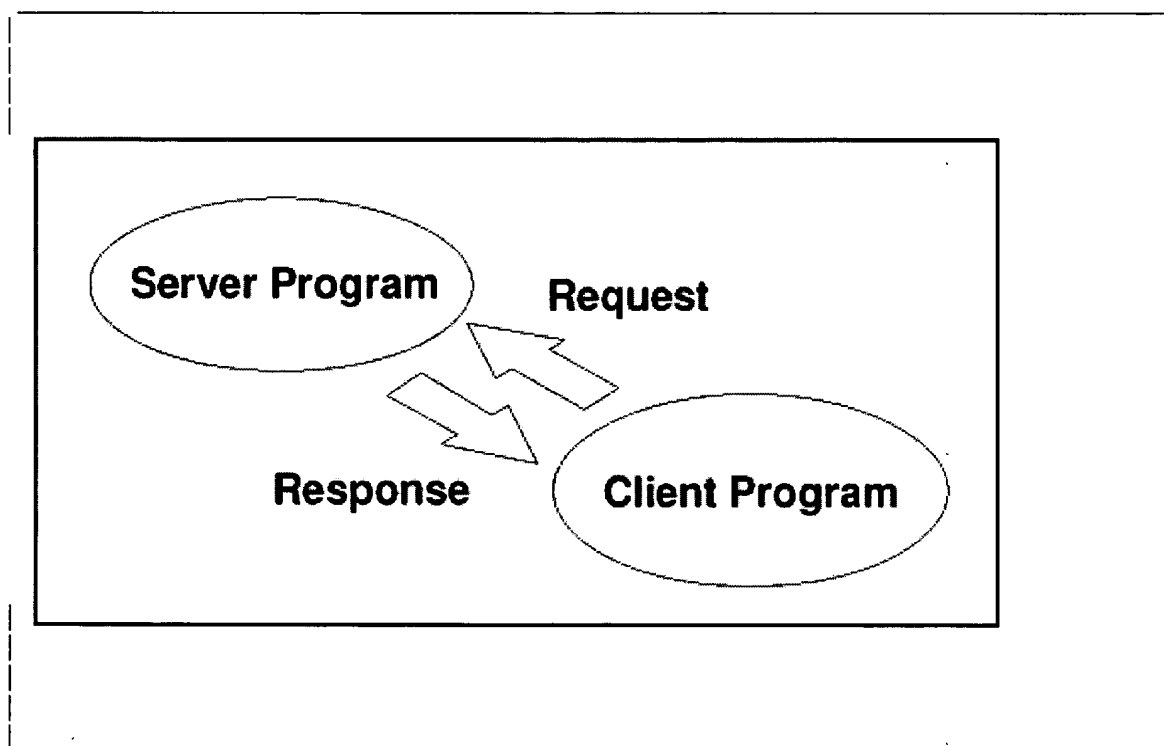


Figure 5. Client/Server Model

For example, the distributed file system (DFS) Service offered by DCE is a distributed application based on the client/server model. The client program of the DFS resides on every DCE system. With the underlying help of the DFS client program, you will be able to access files that reside on remote hosts. On remote hosts, the DFS server program is running, and listening for file-access requests sent by client programs on behalf of users. After the DFS server program fulfills the request and sends back a response to the client host, the DFS client program returns the response to the user.



 [Home](#) | [Products & services](#) | [Support & downloads](#) | [My account](#)**Select a country**[← Offering Information](#)**Announcement Letter**[Asia Pacific letters](#)[Canada letters](#)[EMEA letters](#)[Latin America letters](#)[US letters](#)[Feedback](#)

IBM DCE for Windows NT, Version 2.2 for Portable, Integrated Solutions and Applications

Software Announcement

November 17, 1998

Announcement Letter Number: 298-422

Related Link:[Subscribe to IBM e-news](#)**Table of Contents:**

- [At a Glance](#)
- [DESCRIPTION](#)
- [SUPPLEMENTAL INFORMATION](#)
- [EDUCATION SUPPORT](#)
- [PUBLICATIONS](#)
- [TECHNICAL INFORMATION](#)
- [ORDERING INFORMATION](#)
- [TERMS AND CONDITIONS](#)
- [CHARGES](#)
- [CALL NOW TO ORDER](#)

At a Glance

- Reduced disk space and memory with new Slim Client
- Public Key Certificate Login based on TOG Request for Comments (RFC) 68.4
- Support for machines with multiple NICs
- Support for DCE client machines configured with DHCP
- CDS Preferencing
- Kerberos V5 interoperability
- Public Key Server support
- Microsoft Visual C++ V5.0 compiler support
- Performance improvements including better management of RPC and configuration parameters
- Data Encryption Standard (DES) and Commercial Data Masking Facility (CDMF) encryption capabilities
- Year 2000 ready

For ordering, contact:

Your IBM representative, an IBM
Business Partner, or IBM North America
Sales Centers at
800-IBM-CALL Reference: SE010

EXTRA! EXTRA! . . .

Subscribe to IBM iSource, your electronic source for customized IBM information!

Go to our web site at

<http://www.ibm.com/isource>

or send an e-mail to

info@isource.ibm.com

with the word SUBSCRIBE in the body.

Overview

IBM Distributed Computing Environment for Windows NT (R), Version 2.2 (DCE NT V2.2), an extension of DCE for Windows NT V2.0, is a server/client package based on DCE Release 1.2.2 from The Open Group (TOG).

Major new features include:

- Slim Client
- Public Key Certificate Login
- Kerberos V5 interoperability
- Remote procedure call (RPC) code set conversion

Additionally, this program includes support for systems with multiple Network Interface Cards (NICs) and systems configured to use the Dynamic Host Configuration Protocol (DHCP), plus performance improvements and enhanced usability.

Major components of DCE NT V2.2:

- DCE Runtime Services (client services)
- Slim Client
- Cell Directory Services (CDS)
- Security Services (SS)
- Application Development Kit (ADK)

The DCE NT V2.2 program package also includes IBM DCE Application Development Kit and Runtime Services for Windows (TM) 95, Version 2.0, for use on Windows 95 systems in your DCE network. For customers who use software servers other than DCE for Windows NT, two additional program packages are available:

- IBM DCE Runtime Services for Windows NT, Version 2.2 -- enables Windows NT systems to act as DCE clients using either the full administrative client services or the Slim Client.
- IBM DCE Application Development Kit and Runtime Services for Windows NT, Version 2.2 -- contains components for DCE application developers along with the full administrative client services and the Slim Client.

IBM DCE Management for Tivoli Management Framework, Version 1.0, a feature for use in Tivoli environments, is included in each of the program packages in this announcement. This English-only feature is not available as a separate program.

TOG was formerly known as Open Software Foundation (OSF). OSF is still used in some technical descriptions.

Key Prerequisites

- Computers with Intel-based processors on a TCP/IP, LAN, or WAN network and running Microsoft (TM) Windows NT 4.0 with Service Pack 3.
- Entrust/Entelligence 4.0 client is required for Public Key Certificate Login.

Planned Availability Date

November 30, 1998

This announcement is provided for your information only. For additional information, contact your IBM representative, call 800-IBM-4YOU, or visit the IBM home page at: <http://www.ibm.com>

DESCRIPTION

The IBM DCE NT V2.2 programs, jointly developed by IBM and Digital (TM) Equipment Corporation, are based on DCE Release 1.2.2 and serve as upgrades for the DCE for Windows NT Version 2.0 programs based on DCE Release 1.2.1.

New in V2.2:

- Slim Client reduces DCE memory and disk space resource consumption on client systems. The Slim Client provides the same programming environment to RPC-based applications as the full DCE (Runtime Services) but requires no cell administrator intervention for configuration (configuration local only).
- Public Key Certificate Login allows DCE users to prove their identity to the DCE authentication service using an X509v3 digital certificate and its associated public key pair, rather than a shared secret key password. One benefit of this authentication mechanism is that, in the event of a compromise of the DCE Security Server, public key users do not have any identifying information exposed to the intruder. Users need not have either a traditional secret-key password nor a public key pair generated by the DCE Security Server. This feature is intended for customers who are currently using the Entrust Public Key Infrastructure (PKI) and have a need to map Entrust users to DCE users for authentication and access to resources provided by DCE. DCE NT 2.2 servers and clients support this public key certificate login feature.
- DHCP Client support enables DCE NT V2.2 clients to run on Windows NT workstations or servers that use DHCP to obtain their IP addresses. DCE servers must have IP addresses that remain constant.
- Multiple NIC support enables DCE for Windows NT V2.2 to run on PCs with multiple NICs installed. An environment variable is used to determine which of the NICs is used.
- CDS Preferencing improves the performance of CDS clients by providing a ranking to the order in which clearinghouses are contacted by the client for CDS information. This is accomplished automatically through the use of defaults associated with the location of CDS clients with respect to CDS servers or by manual overrides made by cell administrators.
- Kerberos V5 Interoperability is a TOG DCE 1.2.2 feature that includes an implementation of the Kerberos Version 5 (V5) authentication and key distribution service in the DCE Security Server. Kerberos V5 enables applications running on either DCE or non-DCE platforms to access the DCE Security Server as a full-function IETF-RFC Kerberos Server.
- Public Key Server support provides the TOG DCE 1.2.2 capability of using public and private keys for initial DCE authentication from client systems that support the TOG DCE 1.2.2 public key feature. DCE NT 2.2 clients do not support this public key feature. Public key support does not include public key certification API or the private key storage server.
- Microsoft Visual C++ V5.0 compiled DCE applications are fully supported.
- Tunable Timeout Values enable configuration of internal timeout defaults for timeouts and configuration call intervals, Security Server initialization, TCP connections, and calls to CDS.
- RPC Code Set Conversion, enhanced from the original TOG implementation, provides cross-platform code set support. The RPC interface version number has been increased to 2.0, which is the version supported in other IBM implementations of DCE.

For the convenience of customers who have DCE networks that include multiple operating systems, the program package in this announcement includes:

- IBM DCE Application Development Kit and Runtime Services for Windows 95, V2.0
- IBM DCE Management for Tivoli Management Framework, V1.0

Cryptographic Capabilities

Two types of data privacy are offered: Commercial Data Masking Facility (CDMF) and Data Encryption Standard (DES). CDMF enables 40-bit encryption. DES enables both 56-bit and 40-bit.

Both versions are approved for export outside the United States. However, some countries may have import regulations that apply to products containing encryption capability, particularly DES. Contact your representative or your local export/import coordinator for information about your specific location.

Year 2000

These products are Year 2000 ready. When used in accordance with their associated documentation, are capable of correctly processing, providing, and/or receiving date data within and between the twentieth and twenty-first centuries, provided that all products (for example, hardware, software, and firmware) with the products properly exchange accurate date data with them.

The service end date for these Year 2000 ready products is January 31, 2001.

REFERENCE INFORMATION

Product information and software announcements are available via IBM Web sites. Information can be accessed (searched) by product name or number, announcement letter number or date, type of product, or keywords. Visit the following URLs:

<http://www.software.ibm.com/enetwork/dce>

<http://www.ibm.link.ibm.com>

<http://www.ibm.com/news>

Review the following Software Announcements for more details about DCE products:

- [298-087](#) dated March 24, 1998 (DCE Runtime Services and Application Development Kit for Windows 95, Version 2.0)
- [298-063](#) dated February 24, 1998 (IBM DCE for AIX (R), Version 2.2, and Related DCE Products)
- [297-376](#) dated September 16, 1997 (IBM DCE for Windows NT, Version 2.0)

Trademarks

IBMLink and eNetwork are trademarks of International Business Machines Corporation in the United States or other countries or both.

AIX is a registered trademark of International Business Machines Corporation in the United States or other countries or both.

Windows and Microsoft are trademarks of Microsoft Corporation. Windows NT is a registered trademark of Microsoft Corporation. Digital is a trademark of Digital Equipment Corporation. Other company, product, and service names may be trademarks or service marks of others.

SUPPLEMENTAL INFORMATION**EDUCATION SUPPORT**

Call IBM Education and Training at 800-IBM-TEACH (426-8322) for catalogs, schedules, and enrollment.

PUBLICATIONS

Release Notes and Quick Beginnings, in the language of the program package, are shipped with each program. All other product information and documentation for all languages is provided on the CD-ROM with the program software.

TECHNICAL INFORMATION**Specified Operating Environment****Requirements for DCE for Windows NT (R)**

Each DCE NT V2.2 Server, Runtime Services Client (full/admin client), or Slim Client, requires an Intel-based system with Microsoft (TM) Windows NT 4.0 Server or Workstation with Service Pack 3 installed. Memory, disk space, and CPU requirements as shown in the tables below:

Table 1: Memory and CPU Requirements

The numbers in this table reflect the Windows NT operating system requirements.

	Memory	CPU
DCE NT V2.2 Server	Minimum: 32 MB Recommended: 64 MB, or higher	Minimum: Pentium (TM) 90 Recommended: Pentium 166, or higher
DCE NT V2.2 Client (Runtime Svcs)	Minimum: 16 MB Recommended: 32 MB, or higher	Minimum: 486 Recommended: Pentium 90, or higher
DCE NT V2.2 Slim Client (Runtime Svcs)	Minimum: 8 MB* Recommended: 16 MB, or higher	Minimum: 486 Recommended: Pentium 90, or higher
* The Slim Client actually uses less than 1 MB of memory.		

Table 2: Disk Space Requirements for DCE NT V2.2 Components

Program Component	Disk Space Requirements
DCE Runtime Services (Client)	29.5 MB
DCE Application Development Kit	6.7 MB

DCE Security Server	4.1 MB
DCE Cell Directory Server	0.9 MB
Event Management System (EMS)	1.0 MB
Simple Network Management Protocol (SNMP)	2.0 MB
Service Files	20.2 MB
Online Documentation	14.3 MB
Slim Client	4.8 MB

Notes

- Memory requirements for user applications and data are not included.
- Server memory requirements vary with the size and usage of the Security Services registry and CDS directory.
- Disk space consists of installation requirements only. It does not include the following:
 - Paging file
 - Log files, security credential files, or other DCE data files
 - Server Security Services registry or CDS server directory

Additional Software Requirements

- DCE Runtime Services for Windows NT must be installed before installing and using DCE ADI for Windows NT, DCE Cell Directory Server for Windows NT, and DCE Security Server for Windows NT.
- A DCE cell with at least one DCE Cell Directory Server and at least one DCE Security Server for Windows NT clients.
- Suitable compilers and linkers must be installed on your system before you can use the DCE ADI for Windows NT. On Intel platforms, Microsoft Visual C++ Version 4.0, or later, provides a compatible environment.
- Entrust Product Requirements:

Entrust products are required only if you plan to use the Public Key Certificate Login feature. For Windows NT, this feature requires the Entrust/Entelligence Version 4.0 (client).

The Entrust Public Key Infrastructure (PKI) is not required on DCE client systems, but must be available for issuing certificates to users. The recommended level of the Entrust/PKI is Version 4.0.

Requirements for DCE for Windows (TM) 95 Clients

Each Windows 95 client requires an Intel-based system with Microsoft Windows 95 Version 1.0 with Service Pack 1 installed, and the memory, disk space, and CPU requirements as shown in the table below:

Table 3: Requirements for Windows 95 Client Systems

	Disk Space	Memory	CPU
Runtime Services	31 MB, or higher	Minimum: 16 MB	Minimum: 486 Recommended: Pentium 90, or higher
Application Development Kit	6.8 MB, or higher	Minimum: 16 MB	Minimum: 486 Recommended: Pentium 90,

			or higher
Additional Documentation	15.7 MB, or higher	Minimum: 16 MB	Minimum: 486 Recommended: Pentium 90, or higher
Service Files	14 MB, or higher	Minimum: 16 MB	Minimum: 486 Recommended: Pentium 90, or higher

Note: Memory and fixed disk requirements for user applications, swapper files, and data are not included.

Additional Software Requirements

- DCE Runtime Services for Windows 95 must be installed before installing and using DCE ADK for Windows 95.
- DCOM for Windows 95, Version 1.1, or later.
- A DCE cell with at least one DCE Cell Directory Server and at least one DCE Security Server.
- Suitable compiler and linkers must be installed on your system before you can use the ADK.
- Microsoft C++ Version 4.0, or later, provides a compatible environment.

For additional information about DCE for Windows 95 programs, refer to Software Announcement 298 dated March 24, 1998 (IBM DCE Runtime Services and Application Development Kit for Windows 95, Version 2.0).

Requirements for DCE Management for Tivoli Management Framework

This IBM feature is for use in Tivoli Management Framework environments. Specific prerequisite requirements are provided in the documentation included with the program software.

Compatibility: DCE NT V2.2 complies with applicable DCE Release 1.1, Release 1.2.1, and Release 1.2.2 specifications from TOG and interoperates with other IBM software servers (such as AIX (R), OS/2 (R) Warp) and non-IBM DCE implementations (such as HP, Digital, Gradient, and Solaris).

DCE NT V2.2 provides source-level runtime compatibility with DCE systems from other vendors for applications that conform to the TOG DCE Application Environment Specification (AES).

More information on interoperability and compatibility is located in the readme.txt file that is part of the NT V2.2 program.

Supported Transport Protocols: DCE NT V2.2 provides RPC communications over TCP/IP and UDP/IP transport protocols.

Limitations: DCE NT V2.2 includes all of TOG DCE 1.2.2 features except:

Unsupported Services:

- Security
 - Transitive Trust in a cell hierarchy
 - The Public Key Certificate Management API
 - The Private Key Storage server
 - User-to-User Authentication
 - Global Groups.
- Directory
 - Hierarchical Cells
 - Global Directory Service (GDS) is not provided in this release. However, GDS provided by another vendor can exist in the same cell and be used for intercell communications.
- RPC -- Single-threaded RPC

Unsupported Commands:

- Security
 - sec_salvage_db
 - rlogin and rlogind
 - rsh and rshd
- Distributed Time Service -- dtss-graph

Unsupported Subroutines:

- Distributed File System (DFS) APIs
- RPC code set wchar_t functions

Limitations of Supported Services

Public Key Certificate Login, based on TOG RFC 68.4, has the following limitations:

- The kinit command cannot be used to refresh expired DCE credentials unless the DCE password is provided. Using the Entrust user profile and passphrase for this refresh operation is not supported. If the Entrust user profile name and passphrase are synchronized with the DCE principal name and password, this limitation is transparent to the user.
- When multiple Entrust users are mapped to a single DCE principal, the level of detail of DCE functionality such as auditing and access control is reduced. Only the DCE principal information is available and used in audit records and access control checks.
- If the pwd_val_type Extended Registry Attribute (ERA) that requires password strength checks is attached to a DCE principal, these checks are only enforced on the DCE password for that principal. The Entrust PKI establishes a separate set of rules which are enforced on the Entrust passphrase.
- The key management API is used only by applications that use the shared-secret key authentication protocol. Application servers cannot use the public key certificate login protocol.
- When using Generic Security Service Application Programming Interface (GSSAPI), the DCE administrator must set up an account in the DCE registry database for the initiator and the acceptor. The acceptor cannot use Public Key Certificate Login. No restrictions apply to the account for the initiator.

Public Key Login Support (based on DCE 1.2.2) has the following limitations:

- The DCE Security Server supports login requests from DCE clients that support the TOG 1.2.2 public key login protocol. The TOG 1.2.2 protocol uses public-private key pairs that are generated by the DCE Security Server itself. This feature is separate from the IBM Public Key Certificate Login for DCE that supports login requests based on public key information that is generated by the client's public key infrastructure.
- The DCE client does not support the use of TOG 1.2.2 public key protocol to login to DCE. For compatibility and interoperability purposes, the DCE Security Server supports these login requests from other DCE clients that do use the protocol.

Planning Information

Direct Customer Support: Direct customer support is available through the Personal Systems Support Line. This fee service enhances your productivity by providing voice and electronic access into the IBM support organization. Personal Systems Support Line will help answer questions pertaining to usage, and software defects for eligible products. For more information call 800-237-5511.

Packaging: Each IBM DCE for Windows NT, Version 2.2, program package contains:

- Program software, including softcopy documentation, on three CD-ROMs:
 1. DCE for Windows NT V2.2
 2. DCE for ADK and Runtime Services for Windows 95 V2.0
 3. DCE Management for Tivoli Management Framework V1.0
- International Program License Agreement (IPLA)
- License Information (LI)
- IBM Proof of Entitlement (PoE)

- Customer Service and Support Flyer
- Quick Beginnings
- Release Notes

Each IBM DCE Runtime Services for Windows NT, Version 2.2, program package contains:

- Program software, including softcopy documentation, on two CD-ROMs:
 1. DCE Runtime Services for Windows NT V2.2
 2. DCE Management for Tivoli Management Framework, V1.0

- IPLA
- LI
- IBM PoE
- Customer Service and Support Flyer
- Quick Beginnings
- Release Notes

Each IBM DCE ADK and Runtime Services for Windows NT, Version 2.2, program package contains:

- Program software, including softcopy documentation, on two CD-ROMs:
 1. DCE ADK and Runtime Services for Windows NT V2.2
 2. DCE Management for Tivoli Management Framework, V1.0
- IPLA
- LI
- IBM PoE
- Customer Service and Support Flyer
- Quick Beginnings
- Release Notes

Security, Auditability, and Control

The DCE NT V2.2 programs use the security and auditability features through the full support of DCE authenticated RPC, allowing secure access in a distributed computing environment.

The customer is responsible for evaluation, selection, and implementation of security features, administrative procedures, and appropriate controls in application systems and communication facilities.

ORDERING INFORMATION

Program packages with media, use authorizations (UAs) without media, upgrade options, and Passport Advantage options are available for the programs in this announcement.

- IBM DCE for Windows NT, Version 2.2 is a server/client program with multiple components. Purchase of this package includes entitlement for:
 - One installation of the Security Server
 - One installation of the Cell Directory Server
 - One Registered User
 - Unlimited installations of the Application Development Kit
 - Unlimited installations of the Runtime Services or Slim Client on machines that access IBM DCE server

If you wish to install one or both server components on additional machines, you should purchase applicable Install UA in quantities equal to the number of installations.

For additional users defined to the network, you should purchase Registered User UAs in the quantities equal to the number of users defined in your network.

If you wish to install the Runtime Services or Slim Client on systems that access a non-IBM DCE server, you should purchase a Runtime Services install UA for each installation. Refer to the ordering information for DCE Runtime Service for Windows NT.

- IBM DCE Runtime Services for Windows NT, Version 2.2 provides the client components of the NT V2.2 in a separate package. This package is intended primarily for customers who use DCE servers other than Windows NT but want to include Windows NT systems as clients in their network. Purchase of this package entitles you to install either the Runtime Services (full/admin client) or the Slim Client.

Slim Client on one machine. If you wish to install either of these components on additional client machines, you should purchase an Install UA for each installation of either the Runtime Services or the Slim Client.

- IBM DCE ADK and Runtime Services for Windows NT, Version 2.2 includes both the client components and the ADK components of DCE NT V2.2. This package is intended for customers who want to develop or enable distributed applications for Windows NT.

Purchase of this package includes entitlement for one installation of the ADK and one installation of either the Runtime Services (full/admin client) or the Slim Client.

Note: The ADK requires that the Runtime Services be installed prior to installing the ADK.

If you wish to use the ADK and Runtime Services on additional systems, you should purchase an Install UA for each additional machine on which these components are to be installed.

Passport Advantage Program: The Passport Advantage options for these products are maintained by International Business Machines Corporation (R). For more information, visit:

<http://www.lotus.com/passport>

Programs and Use Authorizations

Description	Order Number	Feature Number	Part Number
DCE NT V2.2 Program Package			
DES English	5801-AAR 4341		39L7835
CDMF English	5801-AAR 4342		39L7836

Note: Use Authorizations are for both CDMF and DES versions

DCE NT V2.2 UAs:

1 Server Install	5802-AAR	2788	39L7879
1 Registered User	5807-AAR	1217	39L7881
5 Registered Users	5807-AAR	1218	39L7882
10 Registered Users	5807-AAR	1219	39L7883
50 Registered Users	5807-AAR	1222	39L7884

DCE NT V2.2 Optional

Component UAs:

1 Install SS Only	5802-AAR	2789	39L8089
1 Install CDS Only	5802-AAR	2790	39L8090

DCE NT Runtime V2.2

Program Package	5801-AAR	4436	39L7949
DES English			
CDMF English	5801-AAR	4448	39L7950
1 Install UA (for both DES and CDMF versions)	5802-AAR	2791	39L7993

DCE NT ADK/RT V2.2

Program Package	5801-AAR	4491	39L8019
DES English			
CDMF English	5801-AAR	4492	39L8020
1 Install UA (for both DES and CDMF versions)	5802-AAR	2792	39L8063

Upgrades

DCE NT V2.2 Upgrade

Program Package	5803-AAR	1773	39L7857
DES English			
CDMF English	5803-AAR	1774	39L7858

DCE NT V2.2 Upgrade UAs:

1 Server Install Upgrade	5804-AAR	1038	39L7880
--------------------------	----------	------	---------

1 Registered User Upgrade	5808-AAR	0537	39L7885
5 Registered Users Upgrade	5808-AAR	0538	39L7886
10 Registered Users Upgrade	5808-AAR	0539	39L7887
50 Registered Users Upgrade	5808-AAR	0540	39L7888

DCE NT V2.2 Optional

Component UAs:

1 Install Upgrade

SS Only	5804-AAR	1039	39L8091
---------	----------	------	---------

CDS Only	5804-AAR	1040	39L8092
----------	----------	------	---------

DCE NT Runtime V2.2

Upgrade Program Package

DES English	5803-AAR	1715	39L7971
-------------	----------	------	---------

CDMF English	5803-AAR	1716	39L7972
--------------	----------	------	---------

1 Upgrade Install UA	5804-AAR	1041	39L7994
----------------------	----------	------	---------

DCE NT ADK/RT V2.2

Upgrade Program Package

DES English	5803-AAR	1742	39L8041
-------------	----------	------	---------

CDMF English	5803-AAR	1744	39L8042
--------------	----------	------	---------

1 Upgrade Install UA	5804-AAR	1042	39L8064
----------------------	----------	------	---------

Upgrade Protection (Entitled Customers): Customers who have previously acquired Software Advantage Upgrade Protection, and have not migrated to the Passport Advantage Offering as shown in the table will receive automatically their new media pack shortly after general availability.

Software Advantage Upgrade Protection Entitlement

	Version 2.0 Upgrade Protection Part Numbers (English)	Version 2.2 Media Pack Part Number (Select One)
Description		
DCE for Windows NT (DES)		
1 Registered User	4071132 (DES)	39L7889
1 Install SS & CDS	4071133 (CDMF)	39L7890
1 Install of SS Only	4076094	
1 Install of CDS Only	4076100	
DCE Runtime Svcs for Windows NT (DES)		
1 Install	4302144 (DES) 39L7996 (CMF)	39L7995
DCE ADK/RT Svcs for Windows NT (DES)		
1 Install	04L0373 (DES) 38L8066 (CDMF)	39L8065

TERMS AND CONDITIONS

Licensing: IPLA. PoEs are required for all authorized use.

Refer to the Ordering Information section for details about entitlements and use authorizations.

The DCE Management for Tivoli Management Framework feature may be installed on as many machines as the customer needs without UAs or charges. Program Services are available for this program.

Limited Warranty Applies: Yes

Program Services: Available until January 31, 2001

Money-back Guarantee: 30-day, money-back guarantee

Copy and Use on Home/Portable Computer: Yes

Support Line: Personal Systems Support Line

Upgrades: You can acquire upgrades up to the currently authorized level of use of the qualifying program.

Volume Orders: Yes, contact your IBM representative.

Passport Advantage Applies: Yes

AIX/UNIX (R) Upgrade Protection Applies: No

Entitled Upgrade for Current AIX/UNIX Upgrade Protection Licensees: No

Variable Charges Apply: No

CHARGES

The charges provided in this announcement are suggested retail prices for the U.S. only and are provided for your information only. Dealer prices may vary, and prices may also vary by country. Prices are subject to change without notice. For additional information and current prices, contact your local IBM representative.

IBM DCE for Windows NT, Version 2.2

Description	Part Number	OTC
Program Packages and Use Authorizations		
Program Package		
with DES	39L7835	\$3,999
with CDMF	39L7836	3,999
UA for 1 Server Install		
SS & CDS	39L7879	3,969
SS only	39L8089	2,199
CDS only	39L8090	1,799
UA for		
1 Registered User	39L7881	29
5 Registered Users	39L7882	139
10 Registered Users	39L7883	269
50 Registered Users	39L7884	1,335
	Part	
Description	Number	OTC
Upgrade Program Packages and Upgrade UAs		
Upgrade Program Pkg		
with DES	39L7857	\$2,399
with CDMF	39L7858	2,399

Upgrade UA: 1 Server Install		
SS & CDS	39L7880	2,385
SS only	39L8091	1,319
CDS only	39L8092	1,079
Upgrade UA:		
1 Registered User	39L7885	19
5 Registered Users	39L7886	85
10 Registered Users	39L7887	159
50 Registered Users	39L7888	789

IBM DCE Runtime Services for Windows NT, Version 2.2

Description	Part Number	OTC
Program Packages and Use Authorizations		
Program Package		
with DES	39L7949	\$149
with CDMF	39L7950	149
UA: 1 Install	39L7993	95
Upgrade Program Packages and Upgrade UAs		
Upgrade Program Package		
with DES	39L7971	89
with CDMF	39L7972	89
Upgrade UA: 1 Install	39L7994	59

IBM DCE ADK & Runtime Svc for Windows NT, Version 2.2

Description	Part Number	OTC
Program Packages and Use Authorizations		
Program Package		
with DES	39L8019	\$529
with CDMF	39L8020	529
UA: 1 Install	39L8063	499
Upgrade Program Packages and Upgrade UAs		
Upgrade Program Package		
with DES	39L8041	319
with CDMF	39L8042	319
Upgrade UA: 1 Install	39L8064	299

Optional Support Line Charge (per Month): \$210

Note: For Passport Advantage ordering information and charges, contact your IBM Lotus representative authorized IBM Lotus Business Partner. Additional information is also available on the Passport Advantage URL:

<http://www.lotus.com/passportadvantage>

CALL NOW TO ORDER

To order, contact the IBM North America Sales Centers, your local IBM representative, or your IBM Business Partner.

IBM North America Sales Centers, our national direct marketing organization, can add your name to the mailing list for catalogs of IBM products.

Phone: 800-IBM-CALL
Fax: 800-2IBM-FAX
Internet: ibm_direct@vnet.ibm.com
Mail: IBM North America Sales Centers
Dept. SE010
P.O. Box 2690
Atlanta, GA 30301-2690
Reference: SE010

To identify your local IBM Business Partner or IBM representative, call 800-IBM-4YOU.

Note: Shipments will begin after the planned availability date.

Trademarks

AIX and OS/2 are registered trademarks of International Business Machines Corporation in the United States or other countries or both.

Pentium is a trademark of Intel Corporation.

Microsoft and Windows are trademarks of Microsoft Corporation.

Windows NT is a registered trademark of Microsoft Corporation.

UNIX is a registered trademark in the United States and other countries exclusively through X/Open Company Limited.

Lotus is a registered trademark of Lotus Development Corporation.

Other company, product, and service names may be trademarks or service marks of others.

[About IBM](#) | [Privacy](#) | [Terms of use](#) | [Contact](#)



Administration Guide

Table of Contents

- [About This Book](#)
- [Who Should Use This book](#)
- [How This Book is Structured](#)

Part 1. The World of DB2 Universal Database

Chapter 1. Administering DB2 Universal Database

Part 2. Database Concepts

Chapter 2. Basic Relational Database Concepts

- [Overview of Database Objects](#)
- [Instances](#)
- [Databases](#)
- [Nodegroups](#)
- [Tables](#)
- [Views](#)
- [Indexes](#)
- [Schemas](#)
- [System Catalog Tables](#)
- [Overview of Recovery Objects](#)
- [Recovery Log Files](#)
- [Recovery History File](#)
- [Overview of Storage Objects](#)
- [Table Spaces](#)
- [Containers](#)
- [Buffer Pool](#)
- [Overview of System Objects](#)
- [Configuration Parameters](#)
- [Business Rules for Data](#)
- [Recovering a Database](#)
- [Overview of Recovery](#)
- [Factors Affecting Recovery](#)

Administration Guide

DCE Directory Services

DCE is an Open Systems Foundation** (OSF**) architecture that provides tools and services to support the creation, use, and maintenance of applications in a distributed heterogeneous computing environment. It is a layer between the operating system, the network, and a distributed application that allows client applications to access remote servers.

With local directories, the physical location of the target database is individually stored on each client workstation in the database directory and node directory. The database administrator can therefore spend a large amount of time updating and changing these directories. The DCE directory services provide a central directory alternative to the local directories. It allows information about a database or a database manager instance to be recorded once in a central location, and any changes or updates to be made at that one location.

DCE is not a prerequisite for running DB2, but if you are operating in a DCE environment, see [Appendix E, Using Distributed Computing Environment \(DCE\) Directory Services](#) for more information.

[[Top of Page](#) | [Previous Page](#) | [Next Page](#)]



Administration Guide

Table of Contents

- [About This Book](#)
- [Who Should Use This book](#)
- [How This Book is Structured](#)

Part 1. The World of DB2 Universal Database

- [Chapter 1. Administering DB2 Universal Database](#)

Part 2. Database Concepts

- [Chapter 2. Basic Relational Database Concepts](#)

- [Overview of Database](#)

[Objects](#)

- [Instances](#)
- [Databases](#)
- [Nodegroups](#)
- [Tables](#)
- [Views](#)
- [Indexes](#)
- [Schemas](#)
- [System Catalog Tables](#)
- [Overview of Recovery](#)

[Objects](#)

- [Recovery Log Files](#)
- [Recovery History File](#)
- [Overview of Storage Objects](#)

- [Table Spaces](#)
- [Containers](#)
- [Buffer Pool](#)
- [Overview of System Objects](#)
- [Configuration Parameters](#)
- [Business Rules for Data](#)
- [Recovering a Database](#)
- [Overview of Recovery](#)
- [Factors Affecting Recovery](#)

Administration Guide

Lightweight Directory Access Protocol (LDAP) Directory Services

Lightweight Directory Access Protocol (LDAP) is an industry standard access method to directory services. A directory service is a repository of resource information about multiple systems and services within a distributed environment; and it provides client and server access to these resources. Each database server instance will publish its existence to an LDAP server and provide database information to the LDAP directory when the databases are created. When a client connects to a database, the catalog information for the server can be retrieved from the LDAP directory. Each client is no longer required to store catalog information locally on each machine. Client applications search the LDAP directory for information required to connect to the database.

LDAP is not a prerequisite for running DB2, but if you are operating in an LDAP environment, see [Appendix Q, Lightweight Directory Access Protocol \(LDAP\) Directory Services](#) for more information.

[[Top of Page](#) | [Previous Page](#) | [Next Page](#)]



Administration Guide

Table of Contents

- [About This Book](#)
 - [Who Should Use This book](#)
 - [How This Book is Structured](#)

Part 1. The World of DB2 Universal Database

- [Chapter 1. Administering DB2 Universal Database](#)

Part 2. Database Concepts

- [Chapter 2. Basic Relational Database Concepts](#)

- [Overview of Database](#)

Objects

- [Instances](#)
- [Databases](#)
- [Nodegroups](#)

Tables

Views

Indexes

Schemas

System Catalog Tables

Overview of Recovery

Objects

- [Recovery Log Files](#)
- [Recovery History File](#)
- [Overview of Storage Objects](#)

Table Spaces

Containers

Buffer Pool

Overview of System Objects

Configuration Parameters

Business Rules for Data

Recovering a Database

Overview of Recovery

Factors Affecting Recovery

Administration Guide

Creating Nodegroups

You create a nodegroup with the CREATE NODEGROUP statement. This statement specifies the set of nodes on which the table space containers and table data are to reside. This statement also:

- Creates a partitioning map for the nodegroup. For details about the partitioning map, see [Partitioning Maps](#).
- Generates a partitioning map ID.
- Inserts records into the following catalog tables:
 - SYSCAT.NODEGROUPS
 - SYSCAT.PARTITIONMAPS
 - SYSCAT.NODEGROUPDEF

To create a nodegroup using the Control Center:

1. Expand the object tree until you see the **Nodegroups** folder.
2. Right-click the **Nodegroups** folder, and select **Create** from the pop-up menu.
3. On the Create Nodegroups window, complete the information, use the arrows to move nodes from the **Available nodes** box to the **Selected nodes** box, and click **Ok**.

To create a nodegroup using the command line, enter:

```
CREATE NODEGROUP <name> ON NODES (<value>,<value>)
```

Assume that you want to load some tables on a subset of the database partitions in your database. You would use the following command to create a nodegroup of two nodes (1 and 2) in a database consisting of at least three (0 to 2) nodes:

```
CREATE NODEGROUP mixng12 ON NODES (1,2)
```

For more information about creating nodegroups, refer to the *SQL Reference* manual.

The CREATE DATABASE command or sqlecrea() API also create the default system nodegroups, IBMDEFAULTGROUP, IBMCATGROUP, and IBMTEMPGROUP. (See [Designing and Choosing Table Spaces](#) for information.)



Administration Guide

Table of Contents

- [About This Book](#)
- [Who Should Use This book](#)
- [How This Book is Structured](#)

Part 1. The World of DB2 Universal Database

• [Chapter 1. Administering DB2 Universal Database](#)

Part 2. Database Concepts

• [Chapter 2. Basic Relational Database Concepts](#)

- [Overview of Database](#)

[Objects](#)

- [Instances](#)
- [Databases](#)
- [Nodegroups](#)
- [Tables](#)
- [Views](#)
- [Indexes](#)
- [Schemas](#)
- [System Catalog Tables](#)
- [Overview of Recovery](#)

[Objects](#)

- [Recovery Log Files](#)
- [Recovery History File](#)
- [Overview of Storage Objects](#)

- [Table Spaces](#)
- [Containers](#)
- [Buffer Pool](#)
- [Overview of System Objects](#)
- [Configuration Parameters](#)
- [Business Rules for Data](#)
- [Recovering a Database](#)
- [Overview of Recovery](#)
- [Factors Affecting Recovery](#)

Administration Guide

Definition of Database Recovery Log

A *database recovery log* keeps a record of all changes made to a database, including the addition of new tables or updates to existing ones. This log is made up of a number of *log extents*, each contained in a separate file called a *log file*.

The database recovery log can be used to ensure that a failure (for example, a system power outage or application error) does not leave the database in an inconsistent state. In case of a failure, the changes already made but not committed are rolled back, and all committed transactions, which may not have been physically written to disk, are redone. These actions ensure the integrity of the database.

For more information, see [Chapter 19, Recovering a Database](#).

[[Top of Page](#) | [Previous Page](#) | [Next Page](#)]

Administration Guide

Binding Utilities to the Database

When a database is created, the database manager attempts to bind the utilities in `db2ubind.lst` to the database. This file is stored in the `bnd` subdirectory of your `sqllib` directory.

Binding a utility creates a *package*, which is an object that includes all the information needed to process specific SQL statements from a single source file.

Note: If you wish to use these utilities from a client, you must bind them explicitly. Refer to the *Quick Beginnings* manual appropriate to your platform for information.

If for some reason you need to bind or rebind the utilities to a database, issue the following commands using the command line processor:

```
connect to sample
bind @db2ubind.lst
```

Note: You must be in the directory where these files reside to create the packages in the `sample` database. The bind files are found in the `BND` subdirectory of the `SQLLIB` directory. In this example, `sample` is the name of the database.

[[Top of Page](#) | [Previous Page](#) | [Next Page](#)]

Administration Guide

Cataloging a Database

When you create a new database, it is automatically cataloged in the system database directory file. You may also use the CATALOG DATABASE command to explicitly catalog a database in the system database directory file. The CATALOG DATABASE command allows you to catalog a database with a different alias name, or to catalog a database entry that was previously deleted using the UNCATALOG DATABASE command.

The following command line processor command catalogs the person1 database as humanres:

```
catalog database person1 as humanres  
with "Human Resources Database"
```

Here, the system database directory entry will have humanres as the database alias, which is different from the database name (person1).

You can also catalog a database on an instance other than the default. In the following example, connections to database B are to INSTANCE_C.

```
catalog database b as b at node instance_c
```

Note: The CATALOG DATABASE command is also used on client nodes to catalog databases that reside on database server machines. For more information, refer to the *Quick Beginnings* manual appropriate to your platform.

For information on the Distributed Computing Environment (DCE) cell directory, see [DCE Directory Services](#) and [Appendix E, Using Distributed Computing Environment \(DCE\) Directory Services](#).

Note: To improve performance, you may cache directory files, including the database directory, in memory. (See [Directory Cache Support \(dir_cache\)](#) for information about enabling directory caching.) When directory caching is enabled, a change made to a directory (for example, using a CATALOG DATABASE or UNCATALOG DATABASE command) by another application may not become effective until your application is restarted. To refresh the directory cache used by a command line processor session, issue a db2 terminate command.

In addition to the application level cache, a database manager level cache is also used for internal, database manager look-up. To refresh this "shared" cache, issue the db2stop and db2start commands.

See [Directory Cache Support \(dir_cache\)](#) for more information about directory caching.

[[Top of Page](#) | [Previous Page](#) | [Next Page](#)]

Administration Guide

Creating a Table Space

Creating a table space within a database assigns containers to the table space and records its definitions and attributes in the database system catalog. You can then create tables within this table space.

See Designing and Choosing Table Spaces for design information on table spaces.

The syntax of the CREATE TABLESPACE statement is discussed in detail in the *SQL Reference* manual. For information on SMS and DMS table spaces, see Designing and Choosing Table Spaces.

To create a table space using the Control Center:

1. Expand the object tree until you see the **Table spaces** folder.
2. Right-click the **Table spaces** folder, and select **Create --> Table Space Using Wizard** from the pop-up menu.
3. Follow the steps in the wizard to complete your task.

To create an SMS table space using the command line, enter:

```
CREATE TABLESPACE <NAME>
  MANAGED BY SYSTEM
  USING ('<path>')
```

To create an SMS table space using the command line, enter:

```
CREATE TABLESPACE <NAME>
  MANAGED BY DATABASE
  USING (FILE'<path>' <size>)
```

The following SQL statement creates an SMS table space on OS/2 or Windows NT using three directories on three separate drives:

```
CREATE TABLESPACE RESOURCE
  MANAGED BY SYSTEM
  USING ('d:\acc_tbsp', 'e:\acc_tbsp', 'f:\acc_tbsp')
```

The following SQL statement creates a DMS table space on OS/2 using two file containers each with 5,000 pages:

```
CREATE TABLESPACE RESOURCE
  MANAGED BY DATABASE
  USING (FILE'd:\db2data\acc_tbsp' 5000,
        FILE'e:\db2data\acc_tbsp' 5000)
```

In the above two examples, explicit names have been provided for the containers. However, if you specify relative container names, the container is created in the subdirectory created for the database (see Database Directories).

In addition, if part of the path name specified does not exist, the database manager creates it. If a subdirectory is

created by the database manager, it may also be deleted by the database manager when the table space is dropped.

The assumption in the above examples is that the table spaces are not associated with a specific nodegroup. The default nodegroup IBMDEFAULTGROUP is used when the following parameter is not specified in the statement:

```
IN nodegroup
```

The following SQL statement creates a DMS table space on a UNIX-based system using three logical volumes of 10 000 pages each, and specifies their I/O characteristics:

```
CREATE TABLESPACE RESOURCE
  MANAGED BY DATABASE
  USING (DEVICE '/dev/rdblvd6' 10000,
        DEVICE '/dev/rdblvd7' 10000,
        DEVICE '/dev/rdblvd8' 10000)
  OVERHEAD 24.1
  TRANSFERRATE 0.9
```

The UNIX devices mentioned in this SQL statement must already exist, and the instance owner and the SYSADM group must be able to write to them.

The following example creates a DMS table space on a nodegroup called ODDNODEGROUP in a UNIX partitioned database. ODDNODEGROUP must be previously created with a CREATE NODEGROUP statement. In this case, the ODDNODEGROUP nodegroup is assumed to be made up of database partitions numbered 1, 3, and 5. On all database partitions, use the device /dev/hdisk0 for 10 000 4 KB pages. In addition, declare a device for each database partition of 40 000 4 KB pages.

```
CREATE TABLESPACE PLANS
  MANAGED BY DATABASE
  USING (DEVICE '/dev/HDISK0' 10000, DEVICE '/dev/n1hd01' 40000) ON NODE 1
        (DEVICE '/dev/HDISK0' 10000, DEVICE '/dev/n3hd03' 40000) ON NODE 3
        (DEVICE '/dev/HDISK0' 10000, DEVICE '/dev/n5hd05' 40000) ON NODE 5
```

UNIX devices are classified into two categories: character serial devices and block-structured devices. For all file-system devices, it is normal to have a corresponding character serial device (or *raw* device) for each block device (or *cooked* device). The block-structured devices are typically designated by names similar to "hd0" or "fd0". The character serial devices are typically designated by names similar to "rhd0", "rfd0", or "rmt0". These character serial devices have faster access than block devices. The character serial device names should be used on the CREATE TABLESPACE command and not block device names.

The overhead and transfer rate help to determine the best access path to use when the SQL statement is compiled. See [Chapter 22, Application Considerations](#) for information on the OVERHEAD and TRANSFERRATE parameters.

DB2 can greatly improve the performance of sequential I/O using the sequential prefetch facility, which uses parallel I/O. See [Understanding Sequential Prefetching](#) for details on this facility.

You can also create a table space that uses a page size larger than the default 4 KB size. The following SQL statement creates an SMS table space on a UNIX-based system with an 8 KB page size.

```
CREATE TABLESPACE SMS8K
  PAGESIZE 8192
  MANAGED BY SYSTEM
  USING ('FSMS_8K_1')
```

BUFFERPOOL BUFFPOOL8K

Notice that the associated buffer pool must also have the same 8 KB page size.

The created table space cannot be used until the buffer pool it references is activated.

The ALTER TABLESPACE SQL statement can be used to add a container to a DMS table space and modify the PREFETCHSIZE, OVERHEAD, and TRANSFERRATE settings for a table space. The transaction issuing the table space statement should be committed as soon as possible, to prevent system catalog contention.

Note: The PREFETCHSIZE should be a multiple of the EXTENTSIZE. For example if the EXTENTSIZE is 10, the PREFETCHSIZE should be 20 or 30. For more information, See Understanding Sequential Prefetching for more information.

Creating a System Temporary Table Space

A system temporary table space is used to store system temporary tables. When a database is created, one of the three default table spaces defined is a system temporary table space called "TEMPSPACE1".

Note: A database must always have at least one system temporary table space since system temporary tables can only be stored in such a table space.

You can use the CREATE TABLESPACE statement to create another system temporary table space. For example,

```
CREATE SYSTEM TEMPORARY TABLESPACE tmp_tbsp
MANAGED BY SYSTEM
USING ('d:\tmp_tbsp', 'e:\tmp_tbsp')
```

The only nodegroup that can be specified when creating a system temporary table space is IBMTEMPGROUP.

Creating a User Temporary Table Space

A user temporary table space is used to store declared temporary tables.

You can use the CREATE TABLESPACE statement to create a user temporary table space:

```
CREATE USER TEMPORARY TABLESPACE usr_tbsp
MANAGED BY DATABASE
USING (FILE 'd:\db2data\user_tbsp' 5000,
FILE 'e:\db2data\user_tbsp' 5000)
```

Like regular table spaces, user temporary table spaces may be created in any nodegroup other than IBMTEMPGROUP. The default nodegroup used when creating a user temporary table space is IBMDEFAULTGROUP.

The DECLARE GLOBAL TEMPORARY TABLE statement defines declared temporary tables for use within a user temporary table space.

Creating Table Spaces in Nodegroups

By placing a table space in a multiple database partition nodegroup, all of the tables within the table space are divided or partitioned across each database partition in the nodegroup. The table space is created into a nodegroup. Once in a nodegroup, the table space must remain there; it cannot be changed to another nodegroup. The CREATE TABLESPACE statement is used to associate a table space with a nodegroup.

Raw I/O

DB2 Universal Database supports direct disk access (raw I/O). This allows you to attach a direct disk access (raw) device to any DB2 Universal Database system. (The only exceptions are the Linux, Windows 95, and Windows 98 operating systems.) The following list demonstrates the physical and logical methods for identifying this type of device:

- On Windows, to specify a physical hard drive, use the following syntax:
`\\.\PhysicalDriveN`

where N represents one of the physical drives in the system. In this case, N could be replaced by 0, 1, 2, or any other positive integer:

`\\.\PhysicalDisk5`

- On Windows, to specify a logical raw partition (that is, an unformatted partition) use the following syntax:
`\\.\N:`

where N: represents a logical drive letter in the system. For example, N: could be replaced by E: or any other drive letter.

- **Note:** You must have Windows NT Version 4.0 with Service Pack 3 installed to be able to write logs to a device.
- On UNIX-based platforms, use the character serial device name; for example, `/dev/rhd0`

[[Top of Page](#) | [Previous Page](#) | [Next Page](#)]

Administration Guide

Creating a Trigger

A trigger defines a set of actions that are executed in conjunction with, or triggered by, an INSERT, UPDATE, or DELETE clause on a specified base table or a typed table. Some uses of triggers are to:

- Validate input data
- Generate a value for a newly-inserted row
- Read from other tables for cross-referencing purposes
- Write to other tables for audit-trail purposes

You cannot use triggers with nicknames.

You can use triggers to support general forms of integrity or business rules. For example, a trigger can check a customer's credit limit before an order is accepted or update a summary data table.

The benefits of using a trigger are:

- Faster application development: Because a trigger is stored in the database, you do not have to code the actions it does in every application.
- Easier maintenance: Once a trigger is defined, it is automatically invoked when the table that it is created on is accessed.
- Global enforcement of business rules: If a business policy changes, you only need to change the trigger and not each application program.

To create a trigger using the Control Center:

1. Expand the object tree until you see the **Triggers** folder.
2. Right-click the **Triggers** folder, and select **Create** from the pop-up menu.
3. Specify information for the trigger.
4. Specify the action that you want the trigger to invoke, and click **Ok**.

To create a trigger using the command line, enter:

```
CREATE TRIGGER <name>  
  <action> ON <table_name>  
  <operation>  
  <triggered_action>
```

The following SQL statement creates a trigger that increases the number of employees each time a new person is hired, by adding 1 to the number of employees (NBEMP) column in the COMPANY_STATS table each time a row is added to the EMPLOYEE table.

```
CREATE TRIGGER NEW_HIRED  
  AFTER INSERT ON EMPLOYEE  
  FOR EACH ROW MODE DB2SQL  
  UPDATE COMPANY_STATS SET NBEMP = NBEMP+1;
```

A trigger body can include one or more of the following SQL statements: INSERT, searched UPDATE,

searched DELETE, full-selects, SET transition-variable, and SIGNAL SQLSTATE. The trigger can be activated before or after the INSERT, UPDATE, or DELETE statement to which it refers. Refer to the *SQL Reference* for complete syntax information on the CREATE TRIGGER statement. Refer to the *Application Development Guide* for information about creating and using triggers.

Note: If the trigger is a BEFORE trigger, the column name specified by the triggered action may not be a generated column other than the identity column. That is, the generated identity value is visible to BEFORE triggers.

Trigger Dependencies


All dependencies of a trigger on some other object are recorded in the SYSCAT.TRIGDEP catalog. A trigger can depend on many objects. These objects and the dependent trigger are presented in detail in the *SQL Reference* discussion on the DROP statement.

If one of these objects is dropped, the trigger becomes inoperative but its definition is retained in the catalog. To revalidate this trigger, you must retrieve its definition from the catalog and submit a new CREATE TRIGGER statement.

If a trigger is dropped, its description is deleted from the SYSCAT.TRIGGERS catalog view and all of its dependencies are deleted from the SYSCAT.TRIGDEP catalog view. All packages having UPDATE, INSERT, or DELETE dependencies on the trigger are invalidated.


If the dependent object is a view and it is made inoperative, the trigger is also marked inoperative. Any packages dependent on triggers that have been marked inoperative are invalidated. (For more information, see Statement Dependencies When Changing Objects.)


[[Top of Page](#) | [Previous Page](#) | [Next Page](#)]

 [DB2 Information](#)

 [Search](#)

 [Index](#)

 [Glossary](#)

 [Feedback](#)

 [Help](#)



Administration Guide

About This Book

This book provides information necessary to use and administer the year 2000 ready, DB2* relational database management system (RDBMS) products, and includes:

- Information about designing, implementing and managing databases
- Information about configuring and tuning your database environment to improve performance.

Many of the tasks described in this book can be performed using different interfaces:

- The **Command Processor**, which allows you to access and manipulate databases from a graphical interface. From this interface, you can also execute SQL statements and DB2 utility functions. Most examples in this book illustrate the use of this interface. For more information about using the command processor, see the *Command Reference*.
- The **application programming interface**, which allows you to execute DB2 utility functions within an application program. For more information about using the application programming interface, see the *Administrative API Reference*.
- The **Control Center**, which allows you to graphically perform administrative tasks such as configuring the system, managing directories, backing up and recovering the system, scheduling jobs, and managing media. The Control Center also contains Replication Administration to graphically set up the replication of data between systems. Further, the Control Center allows you to execute DB2 utility functions through a graphical user interface. There are different methods to invoke the Control Center depending on your platform. For example, use the `db2cc` command on a command line, (on OS/2) select the Control Center icon from the DB2 folder, or use start panels on Windows platforms. For introductory help, select **Getting started** from the **Help** pull-down of the Control Center window. The **Visual Explain** and **Performance Monitor** tools are invoked from the Control Center.

There are other tools that you can use to perform administration tasks. They include:

- The **Script Center** to store small applications called scripts. These scripts may contain SQL statements, DB2 commands, as well as operating system commands.
- The **Alert Center** to monitor the messages that result from other DB2 operations.
- The **Tool Settings** to change the settings for the Control Center, Alert Center, and Replication.
- The **Journal** to schedule jobs that are to run unattended.
- The **Data Warehouse Center** to manage warehouse objects.

[[Top of Page](#) | [Previous Page](#) | [Next Page](#)]

Administration Guide

Table of Contents

- **About This Book**
 - Who Should Use This book
 - How This Book is Structured
-

Part 1. The World of DB2 Universal Database

- **Chapter 1. Administering DB2 Universal Database**
-

Part 2. Database Concepts

- **Chapter 2. Basic Relational Database Concepts**
 - Overview of Database Objects
 - Instances
 - Databases
 - Nodegroups
 - Tables
 - Views
 - Indexes
 - Schemas
 - System Catalog Tables
 - Overview of Recovery Objects
 - Recovery Log Files
 - Recovery History File
 - Overview of Storage Objects
 - Table Spaces
 - Containers
 - Buffer Pool
 - Overview of System Objects
 - Configuration Parameters
 - Business Rules for Data
 - Recovering a Database
 - Overview of Recovery
 - Factors Affecting Recovery
 - Disaster Recovery Considerations
 - Reducing the Impact of Media Failure
 - Reducing the Impact of Transaction Failure
 - System Clock Synchronization in a Partitioned Database System
 - Reorganizing Tables in a Database
 - Overview of DB2 Security
 - Authentication
 - Authorization
 - Federated Database Authentication and Authorization Overview
- **Chapter 3. Federated Systems**
 - Enabling a Federated System

- **Chapter 4. Parallel Database Systems**

- Nodegroups and Data Partitioning
- Types of Parallelism
- I/O Parallelism
- Query Parallelism
- Utility Parallelism
- Hardware Environments
- Single Partition on a Single Processor
- Single Partition with Multiple Processors
- Multiple Partition Configurations
- Summary of Parallelism Best Suited to Each Hardware Environment

- **Chapter 5. About Data Warehousing**

- What is Data Warehousing?
- Subject Areas
- Warehouse Sources
- Warehouse Targets
- Warehouse Agents and Agent Sites
- Steps and Processes
- Warehousing Tasks

- **Chapter 6. About Spatial Extender**

- The Purpose of Spatial Extender
 - Data that Represents Geographic Features
 - How Data Represents Geographic Features
 - The Nature of Spatial Data
 - Where Spatial Data Comes From
-

Part 3. Database Design

- **Chapter 7. Logical Database Design**

- Decide What Data to Record in the Database
- Define Tables for Each Type of Relationship
- One-to-Many and Many-to-One Relationships
- Many-to-Many Relationships
- One-to-One Relationships
- Provide Column Definitions for All Tables
- Identify One or More Columns as the Primary Key
- Identifying Candidate Key Columns
- Defining Identity Columns
- Ensure that Equal Values Represent the Same Entity
- Consider Normalizing Your Tables
- First Normal Form
- Second Normal Form
- Third Normal Form
- Fourth Normal Form
- Planning for Constraints Enforcement
- Unique Constraints
- Referential Integrity
- Table Check Constraints
- Triggers
- Other Database Design Considerations

- **Chapter 8. Physical Database Design**

- Database Directories
- Database Files
- Estimating Space Requirements for Tables
- System Catalog Tables
- User Table Data
- Long Field Data
- Large Object (LOB) Data
- Index Space
- Additional Space Requirements
- Log File Space
- Temporary Work Space
- Designing Nodegroups
- Nodegroup Design Considerations
- Designing and Choosing Table Spaces
- System Managed Space
- Database Managed Space Table Space
- Table Space Design Considerations
- Federated Database Design Considerations

- **Chapter 9. Designing Distributed Databases**

- Using a Single Database in a Transaction
- Using Multiple Databases in a Single Transaction
- Updating a Single Database
- Updating Multiple Databases
- Other Configuration Considerations
- Host or AS/400 Applications Accessing a LAN Based DB2 Universal Database Server in a Multisite Update
- Understanding the Two-Phase Commit Process
- Recovering from Problems During Two-Phase Commit
- Resynchronizing Indoubt Transactions if AUTORESTART=OFF

- **Chapter 10. Designing for Transaction Managers**

- X/Open Distributed Transaction Processing Model
- Application Program (AP)
- Transaction Manager (TM)
- Resource Managers (RM)
- Setting Up a Database as a Resource Manager
- xa_open and xa_close Strings Usage
- New xa_open String Format for DB2 Version 7
- TPM and TP_MON_NAME Values
- xa_open String Format for Earlier Versions of DB2
- Updating Host or AS/400 Database Servers
- Database Connection Considerations
- Making a Heuristic Decision
- Security Considerations
- Configuration Considerations
- XA Function Supported
- XA Interface Problem Determination
- Configuring XA Transaction Managers to Use DB2 UDB
- Configuring IBM TXSeries CICS
- Configuring IBM TXSeries Encina
- Configuring BEA Tuxedo
- Configuring Microsoft Transaction Server

- **Chapter 11. Designing for High Availability**
 - Hot Standby
 - Examples
 - Mutual Takeover
 - Examples
 - Reconnecting after a Failover
 - Resources
-

Part 4. Administering Using the Control Center

- **Chapter 12. Administering DB2 Using GUI Tools**
 - Administration Tools
 - Common Tool Features
 - Show SQL and Show Command
 - Show Related
 - Generate DDL
 - Filter
 - Help
 - The Control Center
 - Main Elements of the Control Center
 - Using a Customized Control Center in DB2 for OS/390
 - Systems That Can Be Administered
 - Objects that can be Administered
 - Displaying Systems in the Control Center
 - Managing DB2 for OS/390 Objects
 - Adding DB2 for OS/390 Subsystems
 - Managing Gateway Connections
 - Functions You Can Perform from the Control Center
 - Creating New Objects
 - Working with Existing Objects
 - Locating objects (DB2 for OS/390 only)
 - The Satellite Administration Center
 - The Command Center
 - The Script Center
 - Using an Existing Script with the Script Center
 - Scheduling a Saved Command Script to Run
 - The Journal
 - Working with Jobs
 - The License Center
 - The Alert Center
 - Client Configuration Assistant
 - Performance Monitor
 - Event Monitor
 - Using the Monitor Tools
 - Monitoring Performance at a Point in Time
 - Predefined Monitors
 - Action Required When an Object Appears in the Alert Center
 - Analyzing an Event for a Period of Time
 - Event Analyzer
 - Analyzing SQL Statements
 - Improving Performance of a Query
 - Analyzing a Simple Dynamic SQL Statement
 - Managing Remote Databases

- Managing Users
 - Granting and Revoking Authorities and Privileges
 - Moving Data
 - Managing Storage
 - Estimating Table and Index Size
 - Checking Space Available in a Table Space
 - Adding More Space to a Table Space
 - Troubleshooting
 - Replicating Data
 - Using Lightweight Directory Access Protocol
 - Using a Java Control Center
 - Running the Control Center as a Java Applet
 - Using Your Java-based Tools for Administration
-

Part 5. Implementing Your Design

- **Chapter 13. Before Creating a Database**
 - Prerequisites Before Creating a Database
 - Starting DB2
 - Starting DB2 UDB on Windows NT
 - Using Multiple Instances of the Database Manager
 - Organizing and Grouping Objects by Schema
 - Enabling Parallelism
 - Enabling Data Partitioning
 - Stopping DB2
 - Details on Creating a Database
 - Designing Logical and Physical Database Characteristics
 - Creating an Instance
 - License Management
 - Establish Environment Variables and the Profile Registry
 - DB2 Administration Server (DAS)
 - Create a Node Configuration File
 - Creation of the Database Configuration File
 - Replicating Configuration Information Using Response Files
 - Enable FCM Communications
- **Chapter 14. Creating a Database**
 - Definition of Initial Nodegroups
 - Definition of Initial Table Spaces
 - Definition of System Catalog Tables
 - Definition of Database Directories
 - Local Database Directory
 - System Database Directory
 - Node Directory
 - DCE Directory Services
 - Lightweight Directory Access Protocol (LDAP) Directory Services
 - Creating Nodegroups
 - Definition of Database Recovery Log
 - Binding Utilities to the Database
 - Cataloging a Database
 - Creating a Table Space
 - Creating a System Temporary Table Space
 - Creating a User Temporary Table Space

- Creating Table Spaces in Nodegroups
- Raw I/O
- Creating a Schema
- Setting a Schema
- Creating and Populating a Table
- Large Object (LOB) Column Considerations
- Defining Constraints
- Defining a Generated Column on a New Table
- Creating a User-defined Temporary Table
- Defining an Identity Column on a New Table
- Creating a Typed Table
- Populating a Typed Table
- Hierarchy Table
- Creating a Table in Multiple Table Spaces
- Creating a Table in a Partitioned Database
- Creating a Trigger
- Trigger Dependencies
- Creating a User-Defined Function (UDF) or Method
- Creating a Function Mapping
- Creating a Function Template
- Creating a User-Defined Type (UDT)
- Creating a User-Defined Distinct Type
- Creating a User-Defined Structured Type
- Creating a Type Mapping
- Creating a View
- Creating a Typed View
- Creating a Summary Table
- Creating an Alias
- Creating a Wrapper
- Creating a Server
- Using Server Options to Help Define Data Sources and Facilitate Authentication Processing
- Creating a Nickname
- Referencing Nickname and Data Source Objects
- Working with Nickname and Data Source Objects
- Identifying Existing Nicknames and Data Sources
- Creating an Index, Index Extension, or an Index Specification
- Using an Index
- Using the CREATE INDEX Statement
- Creating a User-Defined Extended Index Type
- Details on Index Maintenance
- Details on Index Searching
- Details on Index Exploitation
- A Scenario for Defining an Index Extension
- **Chapter 15. Altering a Database**
 - Before Altering a Database
 - Changing Logical and Physical Design Characteristics
 - Changing the License Information
 - Changing Instances
 - Changing Environment Variables and the Profile Registry Variables
 - Changing the Node Configuration File
 - Changing the Database Configuration
 - Altering a Database
 - Dropping a Database

- Altering a Nodegroup
 - Altering a Table Space
 - Dropping a Schema
 - Modifying a Table in Both Structure and Content
 - Altering a User-Defined Structured Type
 - Deleting and Updating Rows of a Typed Table
 - Renaming an Existing Table
 - Dropping a Table
 - Dropping a User-Defined Temporary Table
 - Dropping a Trigger
 - Dropping a User-Defined Function (UDF), Type Mapping, or Method
 - Dropping a User-Defined Type (UDT) or Type Mapping
 - Altering or Dropping a View
 - Dropping a Summary Table
 - Dropping a Wrapper
 - Altering or Dropping a Server
 - Altering or Dropping a Nickname
 - Dropping an Index, Index Extension, or an Index Specification
 - Statement Dependencies When Changing Objects
-

Part 6. Database Security

- **Chapter 16. Controlling Database Access**
 - Selecting User IDs and Groups for Your Installation
 - Windows NT Platform Considerations
 - UNIX Platform Considerations
 - General Rules
 - Selecting an Authentication Method for Your Server
 - Authentication Considerations for Remote Clients
 - Partitioned Database Considerations
 - Using DCE Security Services to Authenticate Users
 - How to Setup a DB2 User for DCE
 - How to Setup a DB2 Server to Use DCE
 - How to Setup a DB2 Client Instance to Use DCE
 - DB2 Restrictions Using DCE Security
 - Federated Database Authentication Processing
 - Authentication Settings
 - Passing IDs and Passwords to Data Sources
 - Federated Database Authentication Example
 - Privileges, Authorities, and Authorization
 - System Administration Authority (SYSADM)
 - System Control Authority (SYSCTRL)
 - System Maintenance Authority (SYSMAINT)
 - Database Administration Authority (DBADM)
 - LOAD Authority
 - Database Privileges
 - Schema Privileges
 - Table Space Privileges
 - Table and View Privileges
 - Nickname Privileges
 - Server Privileges
 - Package Privileges
 - Index Privileges

- Controlling Access to Database Objects
 - Granting Privileges
 - Revoking Privileges
 - Managing Implicit Authorizations by Creating and Dropping Objects
 - Establishing Ownership of a Plan or a Package
 - Allowing Indirect Privileges through a Package
 - Allowing Indirect Privileges through a Package Containing Nicknames
 - Controlling Access to Data with Views
 - Monitoring Access to Data Using the Audit Facility
 - Tasks and Required Authorizations
 - Using the System Catalog
 - Retrieving Authorization Names with Granted Privileges
 - Retrieving All Names with DBADM Authority
 - Retrieving Names Authorized to Access a Table
 - Retrieving All Privileges Granted to Users
 - Securing the System Catalog Views
- **Chapter 17. Auditing DB2 Activities**
 - Audit Facility Behavior
 - Audit Facility Usage Scenarios
 - Audit Facility Messages
 - Audit Facility Record Layouts
 - Audit Facility Tips and Techniques
 - Controlling DB2 Audit Facility Activities
-

Part 7. Moving Data

- **Chapter 18. Utilities for Moving Data**
-

Part 8. Recovery

- **Chapter 19. Recovering a Database**
 - Crash Recovery
 - Getting to a Consistent Database
 - Transaction Failure Recovery in a Partitioned Database Environment
 - Identifying the Failed Database Partition Server
 - Recovery Method: Version Recovery
 - Backing Up a Database
 - Restoring a Database
 - Recovery Method: Roll-Forward Recovery
 - Backup Considerations
 - Restore Considerations
 - Rolling Forward Changes in a Database
 - Recovery History File Information
 - Garbage Collection
 - DB2 Data Links Manager Considerations
 - Crash Recovery Considerations
 - Backup Utility Considerations
 - Restore and Rollforward Utility Considerations
 - Restoring Databases from an offline Backup without Rolling Forward
 - Restoring Databases and Table Spaces and Rolling Forward to the End of the Logs
 - Restoring Databases and Table Spaces and Rolling Forward to a Point in Time

- [DB2 Data Links Manager and Recovery Interactions](#)
 - [Removing a Table from the Datalink_Reconcile_Not_Possible State](#)
 - [Reconciling Data Links](#)
 - [Tivoli Storage Manager](#)
 - [Setting up an Tivoli Storage Manager Client for UNIX-Based Platforms](#)
 - [Setting up an Tivoli Storage Manager Client for Other Platforms](#)
 - [Considerations for Using Tivoli Storage Manager](#)
 - [Recovering Indoubt Transactions on the Host](#)
 - [Recovery when DB2 Connect Has the DB2 Syncpoint Manager Configured](#)
 - [Recovery when DB2 Connect Does Not Use the DB2 Syncpoint Manager](#)
-

Part 9. Introduction to Performance

- **[Chapter 20. Elements of Performance](#)**
 - [Tuning Guidelines](#)
 - [Disk Storage](#)
 - [Performance Improvement Process](#)
 - [How Much Can a System be Tuned?](#)
 - [A Less Formal Approach](#)
 - [Putting It All Together](#)
 - **[Chapter 21. Architecture and Processes Overview](#)**
 - [Storage Architecture](#)
 - [Database Directory](#)
 - [Table Spaces](#)
 - [Data Management](#)
 - [Record Identifiers and Pages](#)
 - [Space Management](#)
 - [Index Management](#)
 - [Locking](#)
 - [Logging](#)
 - [What Happens When Updating](#)
 - [Process Model](#)
 - [Memory Model](#)
-

Part 10. Tuning Application Performance

- **[Chapter 22. Application Considerations](#)**
 - [Concurrency](#)
 - [Repeatable Read](#)
 - [Read Stability](#)
 - [Cursor Stability](#)
 - [Uncommitted Read](#)
 - [Choosing the Isolation Level](#)
 - [Specifying the Isolation Level](#)
 - [Declared Temporary Tables and Concurrency](#)
 - [Locking](#)
 - [Attributes of Locks](#)
 - [Locks and Application Performance](#)
 - [Factors Affecting Locking](#)
 - [Declared Temporary Tables and Locking](#)
 - [LOCK TABLE Statement](#)

- CLOSE CURSOR WITH RELEASE
- Summary of Locking Considerations
- Adjusting the Optimization Class
- How Do You Set the Optimization Class?
- How Much Optimization is Necessary?
- Restrictions on Result Sets to Improve Performance
- FOR UPDATE Clause
- FOR READ or FETCH ONLY Clause
- OPTIMIZE FOR n ROWS Clause
- FETCH FIRST n ROWS ONLY Clause
- DECLARE CURSOR WITH HOLD Statement
- Row Blocking
- Tuning Queries
- Using a SELECT-Statement
- Guidelines When Using a SELECT-Statement
- Compound SQL
- Performance Considerations and Character Conversion
- Code Page Conversion
- Extended UNIX Code (EUC) Code Page Support
- Stored Procedures
- Activating a Database
- Parallel Processing of Applications

- **Chapter 23. Environmental Considerations**
 - Configuration Parameters Affecting Query Optimization
 - Nodegroup Impact on Query Optimization
 - Table Space Impact on Query Optimization
 - Indexing Impact on Query Optimization
 - Indexing versus No Indexing
 - Using the Index Advisor
 - Guidelines for Indexing
 - Performance Tips for Administering Indexes
 - Server Options Affecting Federated Database Queries

- **Chapter 24. System Catalog Statistics**
 - Collecting Statistics Using the RUNSTATS Utility
 - The Database Partition Where RUNSTATS is Executed
 - Analyzing Statistics
 - Collecting and Using Distribution Statistics
 - Understanding Distribution Statistics
 - When Should You Use Distribution Statistics?
 - How Many Statistics Should You Keep?
 - How Does the Optimizer Use Distribution Statistics?
 - Collecting and Using Detailed Index Statistics
 - Understanding Detailed Index Statistics
 - When Should You Use Detailed Index Statistics?
 - User Update-Capable Catalog Statistics
 - Rules for Updating Catalog Statistics
 - Rules for Updating Table and Nickname Statistics
 - Rules for Updating Column Statistics
 - Rules for Updating Distribution Statistics for Columns
 - Rules for Updating Index Statistics
 - Updating Statistics for User-Defined Functions
 - Modeling Production Databases

- **Chapter 25. Understanding the SQL Compiler**

- Overview of the SQL Compiler
- Rewrite Query by the SQL Compiler
- Operation Merging
- Example - View Merges
- Example - Subquery to Join Transformations
- Example - Redundant Join Elimination
- Example - Shared Aggregation
- Operation Movement
- Example - DISTINCT Elimination
- Example - General Predicate Pushdown
- Example - Decorrelation
- Predicate Translation
- Example - Addition of Implied Predicates
- Example - OR to IN Transformations
- Accounting for Column Correlation
- Data Access Concepts and Optimization
- Index Scan Concepts
- Relation Scan versus Index Scan
- Predicate Terminology
- Join Concepts
- Replicated Summary Tables
- Join Strategies in a Partitioned Database
- Influence of Sorting on the Optimizer
- Optimization Strategies for Intra-Partition Parallelism
- Parallel Scan Strategies
- Parallel Sort Strategies
- Parallel Temporary Tables
- Parallel Aggregation Strategies
- Parallel Join Strategies
- Automatic Summary Tables
- Federated Database Query Compiler Phases
- Pushdown Analysis
- Remote SQL Generation and Global Optimization

- **Chapter 26. SQL Explain Facility**

- Choosing an Explain Tool
- Using the SQL Explain Facility
- Introductory Concepts for Explain
- Explain Information for Data Objects
- Explain Information for Data Operators
- How Explain Information is Organized
- Explain Instance Information
- Explain Snapshot Information
- Explain Table Information
- Obtaining Explain Data
- Capturing Explain Table Information
- Capturing Explain Snapshot Information
- Guidelines on Using Explain Output
- Visual Explain
- SQL Advise Facility

Part 11. Tuning and Configuring Your System

- **Chapter 27. Operational Performance**
 - How DB2 Uses Memory
 - Setting Parameters That Affect Memory Usage
 - FCM Requirements
 - Managing the Database Buffer Pool
 - Managing Multiple Database Buffer Pools
 - Choosing One or Many Buffer Pools
 - Prefetching Data into the Buffer Pool
 - Understanding Sequential Prefetching
 - Understanding List Prefetching
 - Prefetching and Intra-Partition Parallelism
 - Configuring I/O Servers for Prefetching and Parallel I/O
 - Enabling Parallel I/O
 - Allocating Multiple Pages at a Time
 - Sorting
 - Different Types of Sorting
 - Tuning the Parameters that Affect Sorting
 - Looking for Indicators of Sorting Performance Problems
 - Techniques for Managing Sorting Performance
 - Reorganizing Catalogs and User Tables
 - Online Index Reorganization
 - Avoiding the Need to Reorganize Tables
 - Performance Considerations for DMS Devices
 - Managing Initialization Overhead
 - Database Agents
 - Using the Database System Monitor
 - Extending Memory
- **Chapter 28. Using the Governor**
 - Starting and Stopping the Governor
 - The Governor Daemon
 - Creating the Governor Configuration File
 - Governor Log Files
 - Querying Governor Log Files
 - Running the Governor and Database Manager Performance
- **Chapter 29. Scaling Your Configuration Through Adding Processors**
 - Adding Processors to a Machine
 - Adding Database Partitions to a Partitioned Database System
 - Adding Database Partitions to a Running System
 - Adding Database Partitions to a Stopped System
 - Dropping a Database Partition from a System
- **Chapter 30. Redistributing Data Across Database Partitions**
 - How to Partition Data
 - Adding and Dropping Database Partitions
 - Specifying a Target Partitioning Map
 - How Data Is Redistributed Across Database Partitions
 - How Data Is Redistributed in Tables
 - Recovering From Redistribution Errors
 - Data Redistribution and Other Operations
 - Following Data Redistribution
- **Chapter 31. Benchmark Testing**

- Benchmark Testing Methodology
 - Preparing for Benchmark Testing
 - Creating a Benchmark Program
 - Executing the Benchmark Tests

 - **Chapter 32. Configuring DB2**
 - Tuning Configuration Parameters
 - Database Manager Parameters
 - Database Manager Configuration Parameter Summary
 - Database Parameters
 - Database Configuration Parameter Summary
 - Parameter Details by Function
 - Capacity Management
 - Database Shared Memory
 - Application Shared Memory
 - Agent Private Memory
 - Agent/Application Communication Memory
 - Database Manager Instance Memory
 - Locks
 - I/O and Storage
 - Agents
 - Database Application Remote Interface (DARI)
 - Logging and Recovery
 - Database Log Files
 - Database Log Activity
 - Recovery
 - Distributed Unit of Work Recovery
 - Database Management
 - Query Enabler
 - Attributes
 - DB2 Data Links Manager
 - Status
 - Compiler Settings
 - Communications
 - Communication Protocol Setup
 - Distributed Services
 - DB2 Discovery
 - Parallel
 - Communications
 - Parallel Processing
 - Instance Management
 - Diagnostic
 - Database System Monitor Parameters
 - System Management
 - Instance Administration
-

Part 12. High Availability

- **Chapter 33. High Availability Cluster Multi-processing, Enhanced Scalability (HACMP ES) for AIX**
 - Cluster Configuration
 - Configuring a DB2 Database Partition
 - Example of a Hot Standby Configuration
 - Example of a Mutual Takeover Configuration

- Configuration of an NFS Server Node
- Example of an NFS Server Takeover Configuration
- Considerations When Configuring the SP Switch
- DB2 HACMP Configuration Examples
- DB2 HACMP Startup Recommendations
- HACMP ES Event Monitoring and User-defined Events
- HACMP ES Script Files
- DB2 Recovery Script Operations with HACMP ES
- Other Script Utilities
- Monitoring HACMP Clusters
- DB2 SP HACMP ES Installation
- DB2 SP HACMP ES New Installation
- DB2 SP HACMP ES Migration
- DB2 SP HACMP ES Worksheets

- **Chapter 34. High Availability in the Windows NT Environment**
 - Failover Configurations
 - Hot Standby Configuration
 - Mutual Takeover Configuration
 - Using the DB2MSCS Utility
 - Specifying the DB2MSCS.CFG File
 - Setting up Failover for a Single-Partition Database System
 - Setting up a Mutual Takeover Configuration for Two Single-Partition Database Systems
 - Setting up Multiple MSCS Clusters for a Partitioned Database System
 - Maintaining the MSCS System
 - Fallback Considerations
 - Registering Database Drive Mapping for Mutual Takeover Configurations in a Partitioned Database Environment
 - Reconciling the Database Drive Mapping
 - Example - Setting up Two Single-Partition Instances for Mutual Takeover
 - Preliminary Tasks
 - Run the DB2MSCS Utility
 - Example - Setting up a Four-Node Partitioned Database System for Mutual Takeover
 - Preliminary Tasks
 - Run the DB2MSCS Utility
 - Register the Database Drive Mapping for ClusterA
 - Register the Database Drive Mapping for ClusterB
 - Administering DB2 in an MSCS Environment
 - Starting and Stopping DB2 Resources
 - Running Scripts
 - Database Considerations
 - User and Group Support
 - Communications Considerations
 - System Time Considerations
 - Administration Server and Control Center Considerations in a Partitioned Database Environment
 - Limitations and Restrictions

- **Chapter 35. DB2 and High Availability on Sun Cluster 2.2**
 - High Availability
 - Fault Tolerance and Continuous Availability
 - Sun Cluster 2.2
 - Supported Systems
 - Agents
 - Logical Hosts

- [Logical Network Interfaces](#)
 - [Disk Groups and File Systems](#)
 - [Control Methods](#)
 - [Disk and File System Configuration](#)
 - [HA-NFS](#)
 - [The cconsole and ctelnet Utilities](#)
 - [Campus Clustering and Continental Clustering](#)
 - [Common Problems](#)
 - [DB2 Considerations](#)
 - [Applications Connecting to an HA Instance](#)
 - [Disk Layout for EE and EEE Instances](#)
 - [Home Directory Layout for EE and EEE Instances](#)
 - [Logical Hosts and DB2 UDB EEE](#)
 - [DB2 Installation Location and Options](#)
 - [Database and Database Manager Configuration Parameters](#)
 - [Crash Recovery](#)
 - [High Availability through Data Replication](#)
 - [The DB2 High Availability Agent](#)
 - [Registering the hadb2 Service](#)
 - [The hadb2tab File](#)
 - [Control Methods](#)
 - [User Scripts](#)
 - [Other Considerations](#)
 - [Fault Monitor](#)
 - [EEE Considerations](#)
 - [The HA.config File](#)
 - [How Control Methods Run DB2 Commands](#)
 - [Setup](#)
 - [Common Installation Steps](#)
 - [Setup on DB2 UDB Enterprise Edition](#)
 - [Setup on DB2 UDB Enterprise - Extended Edition](#)
 - [The hadb2_setup Command](#)
 - [Failover Time](#)
 - [Troubleshooting](#)
-

Part 13. Appendixes

- **[Appendix A. Naming Rules](#)**
 - [Database Names](#)
 - [Database and Database Alias Names](#)
 - [User IDs and Passwords](#)
 - [Schema Names](#)
 - [Group and User Names](#)
 - [Object Names](#)
 - [Federated Database Object Names](#)
 - [How Case-Sensitive Values Are Preserved in a Federated System](#)
- **[Appendix B. Planning Database Migration](#)**
 - [Migration Considerations](#)
 - [Migration Restrictions](#)
 - [Security and Authorization](#)
 - [Storage Requirements](#)
 - [Release-to-Release Incompatibilities](#)

- Migrating a Database

- **Appendix C. Incompatibilities Between Releases**

- DB2 Universal Database Planned Incompatibilities
- Read-only Views in a Future Version of DB2 Universal Database
- PK_COLNAMES and FK_COLNAMES in a Future Version of DB2 Universal Database
- COLNAMES No Longer Available in a Future Version of DB2 Universal Database
- DB2 Universal Database Version 7 Incompatibilities
- Application Programming
- SQL
- Utilities and Tools
- Connectivity and Coexistence
- DB2 Universal Database Version 6 Incompatibilities
- System Catalog Views
- Application Programming
- SQL
- Database Security and Tuning
- Utilities and Tools
- Connectivity and Coexistence
- Configuration Parameters

- **Appendix D. DB2 Registry and Environment Variables**

- **Appendix E. Using Distributed Computing Environment (DCE) Directory Services**

- Creating Directory Objects
- Database Objects
- Database Locator Objects
- Routing Information Objects
- Attributes of Each Object Class
- Details About Each Attribute
- Directory Services Security
- Configuration Parameters and Registry Variables
- CATALOG and ATTACH Commands, and the CONNECT Statement
- CATALOG GLOBAL DATABASE Command
- CONNECT Statement
- ATTACH Command
- How a Client Connects to a Database
- Connecting to Databases in the Same Cell
- Connecting to a Database in a Different Cell
- How Directories are Searched
- ATTACH Command
- CONNECT Statement
- Temporarily Overriding DCE Directory Information
- Directory Services Tasks
- DCE Administrator Tasks
- Database Administrator Tasks
- Database User Tasks
- Directory Services Restrictions

- **Appendix F. User Exit for Database Recovery**

- Overview for OS/2
- Overview for UNIX-Based Operating Systems
- Invoking a User Exit Program
- Sample User Exit Programs

- Sample User Exit Programs for OS/2
- Sample User Exit Programs for UNIX-Based Operating Systems
- Calling Format
- Calling Format for OS/2
- Calling Format for UNIX-Based or Windows NT Operating Systems
- Archive and Retrieve Considerations
- Backup and Restore Considerations (DB2 for OS/2 only)
- Error Handling

- **Appendix G. Explain Tables and Definitions**
 - EXPLAIN_ARGUMENT Table
 - EXPLAIN_INSTANCE Table
 - EXPLAIN_OBJECT Table
 - EXPLAIN_OPERATOR Table
 - EXPLAIN_PREDICATE Table
 - EXPLAIN_STATEMENT Table
 - EXPLAIN_STREAM Table
 - ADVISE_INDEX Table
 - ADVISE_WORKLOAD Table
 - Table Definitions for Explain Tables
 - EXPLAIN_ARGUMENT Table Definition
 - EXPLAIN_INSTANCE Table Definition
 - EXPLAIN_OBJECT Table Definition
 - EXPLAIN_OPERATOR Table Definition
 - EXPLAIN_PREDICATE Table Definition
 - EXPLAIN_STATEMENT Table Definition
 - EXPLAIN_STREAM Table Definition
 - ADVISE_INDEX Table Definition
 - ADVISE_WORKLOAD Table Definition

- **Appendix H. SQL Explain Tools**
 - Running db2expln and dynexpln
 - db2expln Syntax and Parameters
 - Usage Notes for db2expln
 - dynexpln Syntax and Parameters
 - Usage Notes for dynexpln
 - Description of db2expln and dynexpln Output
 - Table Access
 - Temporary Tables
 - Joins
 - Data Streams
 - Insert, Update, and Delete
 - Row Identifier (RID) Preparation
 - Aggregation
 - Parallel Processing
 - Federated Statement Processing
 - Miscellaneous Statements
 - Examples of db2expln and dynexpln Output
 - Example One: No Parallelism Plan
 - Example Two: Single-Partition Database Plan with Intra-Partition Parallelism
 - Example Three: Multipartition Database Plan with Inter-Partition Parallelism
 - Example Four: Multipartition Database Plan with Inter-Partition and Intra-Partition Parallelism
 - Example Five: Federated Database Plan

- **Appendix I. db2exfmt - Explain Table Format Tool**
- **Appendix J. National Language Support (NLS)**
 - Country Code and Code Page Support
 - Deriving Code Page Values
 - Character Sets
 - Character Set for Identifiers
 - Coding SQL Statements
 - Bidirectional CCSID Support
 - Collating Sequences
 - Datetime Values
 - Unicode/UCS-2 and UTF-8 Support in DB2 UDB
 - Introduction
 - UCS-2/UTF-8 Implementation in DB2 UDB
- **Appendix K. Issuing Commands to Multiple Database Partition Servers**
 - Commands
 - Command Descriptions
 - Specifying the Command to Run
 - Running Commands in Parallel on UNIX-Based Platforms
 - Monitoring rah Processes on UNIX-Based Platforms
 - Additional Rah (Run All Hosts) Information (Solaris and AIX only)
 - Prefix Sequences
 - Specifying the List of Machines
 - Eliminating Duplicate Entries from the List of Machines
 - Controlling the rah Command
 - \$RAHDOTFILES on UNIX-Based Platforms
 - Setting the Default Environment Profile on Windows NT
 - Determining Problems with rah on UNIX-Based Platforms
- **Appendix L. How DB2 for Windows NT Works with Windows NT Security**
 - A Sample Scenario with Server Authentication:
 - A Sample Scenario with Client Authentication and a Windows NT Client Machine:
 - A Sample Scenario with Client Authentication and a Windows 95 Client Machine:
 - Using a Backup Domain Controller with DB2
 - User Authentication with DB2 for Windows NT
 - User Name and Group Name Restrictions
 - DB2 for Windows NT Security Service
 - Installing DB2 on a Backup Domain Controller
 - Authentication With Groups and Domain Security
- **Appendix M. Using the Windows NT Performance Monitor**
 - Registering DB2 with the Windows NT Performance Monitor
 - Enable Remote Access to DB2 Performance Information
 - Displaying DB2 and DB2 Connect Performance Values
 - Accessing Remote DB2 Performance Information
 - Resetting DB2 Performance Values
- **Appendix N. Working with Windows NT or Windows 2000 Database Partition Servers**
 - Listing Database Partition Servers in an Instance
 - Adding a Database Partition Server to an Instance
 - Changing the Database Partition Server Configuration
 - Dropping a Database Partition Server from an Instance

- **Appendix O. Configuring Multiple Logical Nodes**
- **Appendix P. High Speed Inter-node Communications**
 - High Speed Interconnection Using TCP/IP
 - Prerequisites for Using an IBM Netfinity SP Switch
 - High Speed Interconnection Using VI
 - Virtual Interface (VI) Hardware Setup
 - Enabling DB2 to Run Using VI
- **Appendix Q. Lightweight Directory Access Protocol (LDAP) Directory Services**
 - Supporting LDAP Client and Server Configurations
 - Support for Windows 2000 Active Directory
 - Configuring DB2 to Use Active Directory
 - Configuring DB2 in the IBM LDAP Environment
 - Creating an LDAP User
 - Configuring the LDAP User for DB2 Applications
 - Registration of DB2 Servers After Installation
 - Update the Protocol Information for the DB2 Server
 - Catalog a Node Alias for ATTACH
 - Deregistering the DB2 Server
 - Registration of Databases
 - Attaching to a Remote Server
 - Deregistering the Database
 - Refreshing LDAP Entries in Local Database and Node Directories
 - Searching
 - Configure Host Database
 - Setting DB2 Registry Variables at the User Level
 - Enable LDAP Support After Installation is Complete
 - Disable LDAP Support
 - LDAP Support and DB2 Connect
 - Security Considerations
 - Security Considerations for Windows 2000 Active Directory
 - Extending the Directory Schema with DB2 Object Classes and Attributes
 - Extending the Directory Schema for IBM eNetwork Directory Version 2.1
 - Extending the Directory Schema for Windows 2000 Active Directory
 - DB2 Objects in the Windows 2000 Active Directory
 - Object Classes and Attributes Used by DB2
- **Appendix R. Extending the Control Center**
 - Performance Considerations
 - Packaging Considerations
 - Interface Descriptions
 - CCExtension
 - CCObject
 - CCMenuAction
 - CCToolBarAction
 - Usage Scenario
 - MyExtension.java
 - MySample.java
 - MyDatabaseActions.java
 - MyInstance.java
 - MyDB2.java
 - MyDatabases.java
 - MySYSPLAN.java

- [MyTable.java](#)
- [MyDBUser.java](#)
- [MyToolBarAction.java](#)
- [MyAlterAction.java](#)
- [MyAction.java](#)
- [MyDropAction.java](#)
- [MyCascadeAction.java](#)
- [MyCreateAction.java](#)

- **[Appendix S. Using the DB2 Library](#)**

- [DB2 PDF Files and Printed Books](#)
- [DB2 Information](#)
- [Printing the PDF Books](#)
- [Ordering the Printed Books](#)
- [DB2 Online Documentation](#)
- [Accessing Online Help](#)
- [Viewing Information Online](#)
- [Using DB2 Wizards](#)
- [Setting Up a Document Server](#)
- [Searching Information Online](#)

- **[Appendix T. Notices](#)**

- [Trademarks](#)

- **[Index](#)**

- **[Contacting IBM](#)**

- [Product Information](#)